



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

2005-06

Focusing ISAR images using fast adaptive
time-frequency and 3D motion detection on
simulated and experimental radar data

Brinkman, Wade H.

Monterey, California. Naval Postgraduate School

<http://hdl.handle.net/10945/2182>

Downloaded from NPS Archive: Calhoun



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>



NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

**FOCUSING ISAR IMAGES USING FAST ADAPTIVE TIME-
FREQUENCY AND 3D MOTION DETECTION ON SIMULATED
AND EXPERIMENTAL
RADAR DATA**

by

Wade Brinkman

June 2005

Thesis Advisor:
Co-Advisor:
Second Reader:

Michael A. Morgan
T. Thayaparan
Jeffrey B. Knorr

Approved for public release, distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE June 2005	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE: Focusing ISAR Images using Fast Adaptive Time-Frequency and 3D Motion Detection on Simulated and Experimental Radar Data			5. FUNDING NUMBERS	
6. AUTHOR(S) Wade Brinkman				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement (mix case letters)			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) <p>Optimization algorithms were developed for use with the Adaptive Joint Time-Frequency (AJFT) algorithm to reduce Inverse Synthetic Aperture Radar (ISAR) image blurring caused by higher-order target motion. A specific optimization was then applied to 3D motion detection. Evolutionary search methods based on the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) algorithm were designed to rapidly traverse the solution space in order to find the parameters that would bring the ISAR image into focus in the cross-range. 3D motion detection was achieved by using the AJTF PSO to extract the phases of 3 different point scatterers in the target data and measuring their linearity when compared to an ideal phase for the imaging interval under investigation. The algorithms were tested against both simulated and real ISAR data sets.</p>				
14. SUBJECT TERMS Inverse Synthetic Aperture Radar, Adaptive Joint Time-Frequency Algorithm, Genetic Algorithm, Particle Swarm Optimization, 3D Motion Detection, ISAR, AJTF, GA, PSO			15. NUMBER OF PAGES 143	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release, distribution is unlimited

**FOCUSING ISAR IMAGES USING FAST ADAPTIVE TIME-FREQUENCY
AND 3D MOTION DETECTION ON SIMULATED AND EXPERIMENTAL
RADAR DATA**

Wade H. Brinkman
Captain, Canadian Forces
B.S., Royal Military College of Canada, 1997

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

**NAVAL POSTGRADUATE SCHOOL
June 2005**

Author: Wade Brinkman

Approved by: Michael A. Morgan, Thesis Advisor

Thayananthan Thayaparan, DRDC
Co-Advisor

Jeffrey B. Knorr
Second Reader

John P. Powers
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Optimization algorithms were developed for use with the Adaptive Joint Time-Frequency (AJTF) algorithm to reduce Inverse Synthetic Aperture Radar (ISAR) image blurring caused by higher-order target motion. A specific optimization was then applied to 3D motion detection. Evolutionary search methods based on the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) algorithm were designed to rapidly traverse the solution space in order to find the parameters that would bring the ISAR image into focus in the cross-range. 3D motion detection was achieved by using the AJTF PSO to extract the phases of 3 different point scatterers in the target data and measuring their linearity when compared to an ideal phase for the imaging interval under investigation. The algorithms were tested against both simulated and real ISAR data sets.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
A.	RESEARCH REQUIREMENT AND OBJECTIVES.....	1
B.	SCOPE AND ORGANIZATION	2
1.	Scope.....	2
2.	Primary Research Questions	4
3.	Organization.....	4
II.	INTRODUCTION TO INVERSE SYNTHETIC APERTURE RADAR (ISAR) PROCESSING AND THE FOCUSING PROBLEM.....	7
III.	INTRODUCTION TO TIME-FREQUENCY TRANSFORMS AND THE ADAPTIVE JOINT TIME-FREQUENCY ALGORITHM	11
A.	TIME-FREQUENCY TRANSFORMS	11
1.	Short Time Fourier Transform	11
2.	Weaknesses of the STFT and Time – Frequency Transform	15
B.	ADAPTIVE JOINT TIME-FREQUENCY ALGORITHM	16
1.	The Motion Model for a Moving ISAR Target	16
2.	The ISAR Signal and the Motion Compensation Algorithm.....	18
3.	Selection of an Appropriate Basis Function or Phase Function	22
a.	<i>Projection Method to Determine $h_{p_r}(t)$ and $h_{p_r}(t)$</i>	23
b.	<i>Automatic Recognition of Focus using the STFT and the Radon Transform.....</i>	23
c.	<i>Automatic Recognition of Focus using the FFT.....</i>	25
C.	CHAPTER SUMMARY.....	27
IV.	GENETIC ALGORITHM AND PARTICLE SWARM OPTIMIZATION AS SEARCH ALGORITHMS.....	29
A.	INTRODUCTION TO THE SOLUTION SPACE AND THE EXHAUSTIVE SEARCH	29
B.	GENETIC ALGORITHM PARAMETER SEARCH.....	33
1.	Sequence of the Genetic Algorithm.....	34
a.	<i>Define the Population</i>	35
b.	<i>Evaluating Fitness of Each Member of Population.....</i>	36
c.	<i>Generating Children</i>	37
d.	<i>Mutation of the Children.....</i>	39
e.	<i>Evaluating the Fitness of Each Child and Reinsertion of Parents and Children into New Population.....</i>	40
f.	<i>Determine if the Search is Complete.....</i>	42
2.	GA ISAR Results.....	42
a.	<i>B727 Simulated Data Set.....</i>	42
b.	<i>6 – Point Delta Wing Experimental Data Set</i>	48
C.	PARTICLE SWARM OPTIMIZATION PARAMETER SEARCH.....	55
1.	Sequence of the Particle Swarm Optimization Algorithm.....	56

a.	<i>Defining the Particles of the Swarm</i>	56
b.	<i>Assessing Initial Fitness of the Swarm and Defining the Local Best Vector and Global Best Particle</i>	57
c.	<i>Updating the Particle Velocities and Positions</i>	57
d.	<i>Updating Particle Fitness and Reassigning Local Best Vector and Global Best Particle</i>	61
e.	<i>Termination of the Search</i>	61
2.	PSO ISAR Results.....	62
a.	<i>B727 Simulated Data Set</i>	62
b.	<i>6 – Point Delta Wing Experimental Data Set</i>	64
D.	COMPARISON BETWEEN GA AND PSO ISAR FOCUSING ALGORITHMS	68
E.	HIGHER ORDER MOTION COMPENSATION	71
F.	CHAPTER SUMMARY.....	74
V.	3D MOTION DETECTION	75
A.	INTRODUCTION TO THE 3D MOTION MODEL AND THE DETECTION METHOD	75
B.	3D MOTION DETECTION RESULTS	80
1.	Imaging Intervals with 2048 Pulses and 2 nd -Degree Motion Compensation	80
2.	Imaging Intervals with 2048 Pulses and 4 th -Degree Motion Compensation.....	85
3.	Imaging Intervals with 1024 Pulses and 2 nd -Degree Motion Compensation.....	87
4.	Imaging Intervals with 1024 Pulses and 4 th -Degree Motion Compensation.....	92
5.	Comparison between the Different Imaging Interval Sizes and the Degree of Motion Compensation.....	94
C.	CHAPTER SUMMARY.....	96
VI.	CONCLUSION	97
A.	SUMMARY OF RESULTS	97
1.	AJTF GA and PSO Algorithms.....	97
2.	3D Motion Detection	97
B.	FOLLOW-ON WORK.....	98
APPENDIX	MATLAB CODE	99
A.	GA CODE	99
B.	PSO CODE	107
C.	3D MOTION DETECTION CODE	112
	LIST OF REFERENCES.....	119
	INITIAL DISTRIBUTION LIST	121

LIST OF FIGURES

Figure 1.	B727 Simulated Data Set – Unfocused and Focused Images.....	2
Figure 2.	Picture of 6 – Point Delta Wing (From [1].).....	3
Figure 3.	Example of an Unfocused and Focused 6 – Point Delta Wing.....	3
Figure 4.	60,000 Pulse 6 – Point Delta Wing Data Set Focused with the Fast Fourier Transform.....	9
Figure 5.	256 Pulse Imaging Interval 6 – Point Delta Wing Data Set Focused with the Fast Fourier Transform.	9
Figure 6.	ISAR signal, s_{Rx} , of Range Bin 31 and the Hamming Window (After [5].)....	13
Figure 7.	Blurred ISAR Image (close-up of Range Bin 31 in B727 data set), Power Spectrum of Range Bin 31 and its STFT.	13
Figure 8.	Focused ISAR Image (close-up of Range Bin 31 in B727 data set), Power Spectrum of Range Bin 31 and its STFT.	14
Figure 9.	Wigner-Ville Distribution of Unfocused Range Bin 31 (After [3] and [5].) ...	15
Figure 10.	Geometry of the Radar Imaging of a Target (From [3].).....	17
Figure 11.	Unfocused Range Bin (top) and the Best Basis Function $h_{p_r}(t)$ for Translational Motion Compensation.....	20
Figure 12.	Time-Frequency Transform of Range Bin After Translational Motion Compensation.	21
Figure 13.	Radon Transform of Uncompensated and Compensated Range Bins.	25
Figure 14.	Example of a Unfocused and Focused Range Bin and Their Score.	27
Figure 15.	2D and 3D Visualization of B727 Solution Space for Translational Motion Compensation.	30
Figure 16.	2D and 3D Visualization of B727 Solution Space for Rotational Motion Compensation.	31
Figure 17.	Illustration of a GA's Rapid Convergence.....	34
Figure 18.	Flow Chart of the ISAR Genetic Algorithm (After [6].)	35
Figure 19.	Illustration of the Roulette Wheel.....	37
Figure 20.	Illustration of the Mating Pool Selection Method.....	38
Figure 21.	Illustration of the Mutation Process.....	39
Figure 22.	Convergence Graph of GA with Probability of Reinsertion of Children into New Population Equal to 80%.....	41
Figure 23.	Unfocused B727 and the Power Spectras of Range Bins 31 and 44.	43
Figure 24.	B727 and the Power Spectras of Range Bins 31 and 44 after Translational Motion Compensation.....	44
Figure 25.	Focused B727 and the Power Spectras of Range Bins 31 and 44.	44
Figure 26.	Cross-Correlation between a Focused and Unfocused B727 Image.....	45
Figure 27.	Correlation between Two Focused B727 Images.	46
Figure 28.	Percentage of Images Focused Using the Genetic Algorithm as a Function of Population Size and Number of Generations – B727 Data Set.	47
Figure 29.	Average Run Time of the Genetic Algorithm as a Function of Population Size and Number of Generations – B727 Data Set.....	48

Figure 30.	Uncompensated (top) and Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 15 with 2048 Pulses in the Cross-Range – GA.....	50
Figure 31.	Uncompensated (top) and Motion Compensated 6 – Point Delta Wing ISAR Image – Imaging Interval 19 with 2048 Pulse in the Cross-Range – GA.....	51
Figure 32.	Uncompensated (top) and Motion Compensated Delta Wing ISAR Image – Imaging Interval 23 with 2048 Pulses in the Cross-Range – GA.....	52
Figure 33.	Uncompensated (top) and Motion Compensated Delta Wing ISAR Image – Imaging Interval 14 with 2048 Pulses in the Cross-Range – GA.....	53
Figure 34.	Percentage of Images Focused Using the Genetic Algorithm as a Function of Population Size and Number of Generations: 6 – Point Delta Wing Data Set.....	54
Figure 35.	Average Run Time of the Genetic Algorithm as a Function of Population Size and Number of Generations: 6 – Point Delta Wing Data Set.....	55
Figure 36.	The PSO Flowchart (After [7])......	56
Figure 37.	Example of an Initial Swarm in a Solution Space for 4 th -Degree Motion Error.....	57
Figure 38.	Graphical Illustration of the Velocity and Position Update Process. Note that <i>gbest</i> = Global Best and <i>pbest</i> = Local Best (From [7].).....	59
Figure 39.	Example of a Final Swarm in a Solution Space for 4 th -Degree Motion Error.....	62
Figure 40.	B727 Data Set Focused by the PSO Algorithm.....	63
Figure 41.	Percentage of Images Focused Using the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles – B727 Data Set.....	63
Figure 42.	Average Run Time of the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles.....	64
Figure 43.	Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 15 with 2048 Pulses in the Cross-Range – PSO.....	65
Figure 44.	Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 19 with 2048 Pulses in the Cross-Range – PSO.....	65
Figure 45.	Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 23 with 2048 Pulses in the Cross-Range – PSO.....	66
Figure 46.	Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 14 with 2048 Pulses in the Cross-Range – PSO.....	66
Figure 47.	Percentage of Images Focused Using the Particle Swarm Algorithm as a Function of Swarm Size and Number of Cycles – Delta Wing Data Set.....	67
Figure 48.	Average Run Time of the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles – Delta Wing Data Set.....	68
Figure 49.	Combined GA (top) and PSO Convergence Graphs for Selected Population / Swarm Sizes.....	70
Figure 50.	Unfocused Imaging Interval 13.....	72
Figure 51.	Time-Frequency Transforms of Range Bins 25 and 19 – Left: Unfocused Right: Focused.....	73

Figure 52.	Focused Imaging Interval 13.	73
Figure 53.	Image Interval 13 with 2 nd , 3 rd and 4 th -Degree Motion Compensation, Reduced Solution Space.	74
Figure 54.	Engagement of a Target with 3D Motion (From [3].)	76
Figure 55.	Plot of Θ_{R1} as a Function of Θ_{ideal} when no 3D motion is present (After [12].).....	78
Figure 56.	Imaging Interval Possessing a High Degree of Non-Linearity (After [12].) ...	79
Figure 57.	Imaging Interval Possessing a Low Degree of Non-Linearity (After [12].)....	79
Figure 58.	Degree of Non-Linearity of Imaging Intervals of 2048 Pulses and 2 nd - Degree Motion Compensation.	80
Figure 59.	Imaging Interval 23 – Well Focused with Low Degree of Non-Linearity of Phases.....	81
Figure 60.	Imaging Interval 15 – Well Focused with Low Degree of Non-Linearity of Phases.....	82
Figure 61.	Imaging Interval 18 – Well Focused with Low Degree of Non-Linearity of Phases.....	82
Figure 62.	Imaging Interval 11 – Poorly Focused with High Degree of Non-Linearity of Phases.	83
Figure 63.	Imaging Interval 12 – Poorly Focused with High Degree of Non-Linearity of Phases.	83
Figure 64.	Imaging Interval 26 – Poorly Focused with High Degree of Non-Linearity of Phases.	84
Figure 65.	Degree of Non-Linearity of Imaging Intervals of 2048 Pulses and 4 th - Degree Motion Compensation.	85
Figure 66.	Imaging Interval 14 – Well Focused with Low Degree of Non-Linearity of Phases.....	86
Figure 67.	Imaging Interval 28 – Poorly Focused with High Degree of Non-Linearity of Phases.	87
Figure 68.	Degree of Non-Linearity of Imaging Intervals of 1024 Pulses and 2 nd - Degree Motion Compensation.	88
Figure 69.	Imaging Interval 45 – Well Focused with Low Degree of Non-Linearity of Phases.....	89
Figure 70.	Imaging Interval 33 – Well Focused with Low Degree of Non-Linearity of Phases.....	89
Figure 71.	Imaging Interval 35 – Well Focused with Low Degree of Non-Linearity of Phases.....	90
Figure 72.	Imaging Interval 56 – Poorly Focused with High Degree of Non-Linearity of Phases.	90
Figure 73.	Imaging Interval 25 – Poorly Focused with High Degree of Non-Linearity of Phases.	91
Figure 74.	Imaging Interval 29 – Poorly Focused with High Degree of Non-Linearity of Phases.	91
Figure 75.	Degree of Non-Linearity of Imaging Intervals of 1024 Pulses and 4 th - Degree Motion Compensation.	92

Figure 76.	Imaging Interval 10 – Well Focused with Low Degree of Non-Linearity of Phases.....	93
Figure 77.	Imaging Interval 2 – Well Focused with Low Degree of Non-Linearity of Phases.....	93
Figure 78.	Imaging Interval 24 – Poorly Focused with High Degree of Non-Linearity of Phases.	94

LIST OF TABLES

Table 1.	The Genetic Algorithm Population – 2 nd subscript indicated Parent.....	36
Table 2.	Processing Time – Degree of Non-Linearity of Phases for Different Imaging Intervals.	95

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS

NCTR	Non-Cooperative Target Recognition
DRDC	Defence Research and Development Canada
AJTF	Adaptive Joint Time-Frequency
ISAR	Inverse Synthetic Aperture Radar
FFT	Fast Fourier Transform
STFT	Short Time Fourier Transform
WVD	Wigner-Ville Distribution
GA	Genetic Algorithm
PSO	Particle Swarm Optimization
JEM	Jet Engine Modulation
CPI	Coherent Processing Interval
LSF	Least-Squares Fit

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Dr. Thayaparan of DRDC – Ottawa for all of his time and effort he dedicated towards helping me with my thesis. I thoroughly enjoyed our meetings in Ottawa and hope that we will have the opportunity to work together again. I sincerely hope that this work will someday benefit the operational capability of the Canadian Forces.

I would also like to thank Professor Morgan for accepting me as a thesis student regardless of the fact that this area of study was not one of his funded research topics. He allowed me to pursue a topic of both personal interest and of interest to the Canadian Forces.

Most importantly I would like to thank my wife, Ginette, my daughter, Trinity, and my son, Kyler for their selfless sacrifice over the last two years. Their patience and understanding allowed me the freedom to succeed at achieving my Master's Degree.

“Trust in the Lord with all your heart and lean not on your own understanding” - Proverbs 3:5

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Inverse synthetic aperture radar (ISAR) imaging is an effective way to acquire high resolution images of targets of interest at long range and as such is an irreplaceable tool in the task of non-cooperative target recognition (NCTR) of both ships and aircraft. The Canadian Air Force is currently upgrading her fleet of CP-140 Aurora maritime patrol aircraft to possess NCTR through ISAR in order to increase the capability of the Canadian Forces in both sovereignty patrols of Canadian territory and the protection of the Canadian Patrol Frigates and allied ships operating abroad as a coalition. The United States Navy's equivalent aircraft, the P-3 Orion, currently possesses this capability. Therefore, effective ISAR imaging will have a real impact in the decision-making process in future military operations involving both Canadian and American forces.

One significant problem with ISAR image formation is the assumption of time invariance of the Doppler frequency used to resolve the image in the cross-range. Time-variant Doppler becomes present in an ISAR signal when an aircraft is maneuvering or a ship is pitching and rolling during the coherent processing interval and is typically referred to as motion error. Because of this motion error and the fast Fourier transform's (FFT) basic assumption of stationary signals, the use of the FFT to resolve the image in the cross-range will cause extensive blurring and leave the image unrecognizable even to the most experienced ISAR operators.

One proven method of motion compensation is the adaptive joint time-frequency (AJTF) algorithm. However, this algorithm is not without two significant weaknesses of its own. The first problem is that the computational burden of the exhaustive search used to extract the motion compensation parameters is quite large which limits its usefulness in an operational situation. The second problem with the AJTF algorithm is that the target must follow the mathematical model of the AJTF which, in particular, states that rotational motion must be confined to a 2D plane (i.e., no 3D motion).

To address the first problem, two evolutionary searches, a Genetic Algorithm (GA) and a Particle Swarm Optimization (PSO), were investigated, developed and evalu-

ated in order to take advantage of their properties of rapid convergence. Both the GA and the PSO performed excellently during their testing against simulated and real radar data in the extraction of the search parameters from the solution space. They would typically converge to the search parameters required for focusing in less than 50% of the cost (or fitness) function calculations required by the exhaustive search. Also, the PSO proved to be a much more efficient algorithm for this optimization problem with its performance consistently exceeding that of the GA.

To address the second problem, the PSO algorithm designed above was combined with the 3D motion detection algorithm developed at DRDC – Ottawa by Thayaparan. The resulting algorithm was able to effectively and efficiently sort good imaging intervals in the real radar data set from poor imaging intervals which violated the 2D motion assumption of the mathematic model. It was shown that the imaging interval indicated as possessing a low degree of 3D motion created well-focused images after being motion compensated. Imaging intervals that were indicated as possessing a high degree of 3D motion did not focus even after being motion compensated. These imaging intervals can now be discarded instead of presented to the ISAR operator.

This thesis draws from many sources in the radar signal processing community and cites them appropriately. There are two contributions made by this thesis that are novel in their entirety. First, the automatic detection of focus using the FFT for determining basis function suitability in the AJTF algorithm is new, effective and fast and not used in any of the cited references. Secondly, the AJTF PSO algorithm, with the PSO based on the work in [7], is unique to its application to ISAR image focusing. The work on 3D motion detection draws heavily from the work in [12] and provides two significant improvements. First, the measure of non-linearity in an imaging interval is based on the percentage of deviation from the line of least-squares fit as opposed to the basing it on the difference between the line of least-squares fit and the actual line depicting the linearity of phases. Secondly, the AJTF PSO developed in the first part of this thesis is applied to the 3D motion detection algorithm. This creates an algorithm that evaluates the degree of 3D motion in an imaging interval both rapidly and accurately which is useable by an ISAR operator in an operational situation.

I. INTRODUCTION

A. RESEARCH REQUIREMENT AND OBJECTIVES

Inverse synthetic aperture radar (ISAR) imaging is an effective way to acquire high resolution images of targets of interest at long range and as such is an irreplaceable tool in the task of non-cooperative target recognition (NCTR) of both ships and aircraft. The Canadian Air Force is currently upgrading her fleet of CP-140 Aurora maritime patrol aircraft to possess NCTR through ISAR in order to increase the capability of the Canadian Forces in both sovereignty patrols of Canadian territory and the protection of the Canadian Patrol Frigates and allied ships operating abroad as a coalition. The United States Navy's equivalent aircraft, the P-3 Orion, currently possesses this capability. Therefore, effective ISAR imaging will have a real impact in the decision making process in future military operations involving both Canadian and American forces.

As explained in [1], [2] and [3] and repeated in Chapter II of this thesis, one drawback of ISAR is that higher order target motion, observed when ships are pitching and rolling in rough seas or when aircraft are maneuvering, introduces time-varying Doppler into the radar signal. It is this time-varying Doppler that causes severe image blurring in the cross-range during the ISAR signal processing stage and effectively renders the target's image unrecognizable even to the most experienced ISAR operators. Therefore, for ISAR to realize the operational goal of NCTR, methods must be developed to compensate for the higher order motion. The first objective of this thesis was to continue the work by Thayaparan [1] and Chen [3] on the Adaptive Joint Time – Frequency Algorithm (AJTF) and to examine different methods of efficiently removing time-varying Doppler from the radar signal in order to form images that are well focused in the cross-range.

The AJTF algorithm in [1] and [3] assumes 2D motion on a rotational axis during the required ISAR dwell interval. Unfortunately, true targets do not always follow this assumption and within the radar signal there exists 3D motion which cannot be compensated by the AJTF algorithm. One way to deal with the existence of 3D motion in an ISAR imaging data set is to determine to what degree 3D motion exists in an imaging in-

terval of the data set. The imaging intervals with a small degree of 3D motion are imaged while the intervals that possess a high degree of 3D motion are discarded. The research in [3], [4] and [12] provide the background for 3D motion detection using the nonlinearity of phase method. Therefore, the second objective of this thesis is to apply the fast AJTF algorithm found in the first part of this thesis to 3D motion detection methods found in [3], [4] and [12].

B. SCOPE AND ORGANIZATION

1. Scope

This thesis uses two ISAR data sets, one which is a B727 simulated data set created by Chen [18] and is designed to display the ideal motion error found in a blurred ISAR image and is available from the Naval Research Laboratory website. An example of the unfocused and focused image is found in Figure 1. The other ISAR data set, which was used in [1], is the 6 – Point Delta Wing experimental data set collected by Defence Research and Development Canada – Ottawa (DRDC – Ottawa) at their radar range located at Shirley’s Bay, Ontario. A photo of the 6 – Point Delta Wing is shown in Figure 2 and an example of its blurred and focused image can be found in Figure 3.

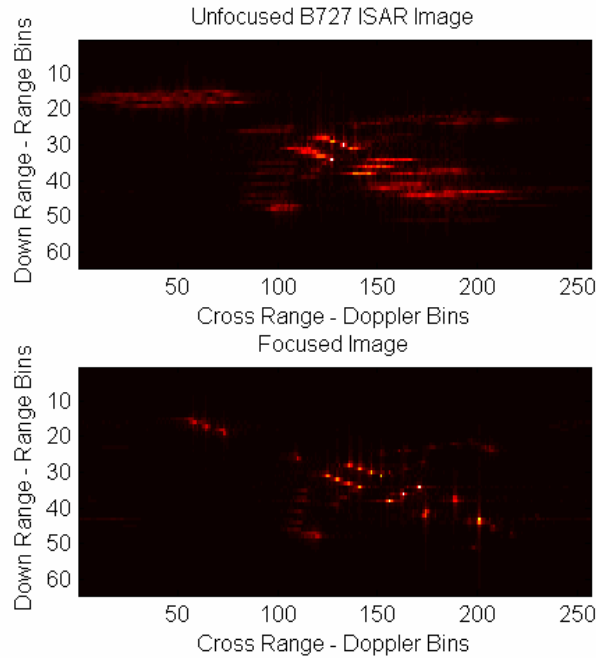


Figure 1. B727 Simulated Data Set – Unfocused and Focused Images.

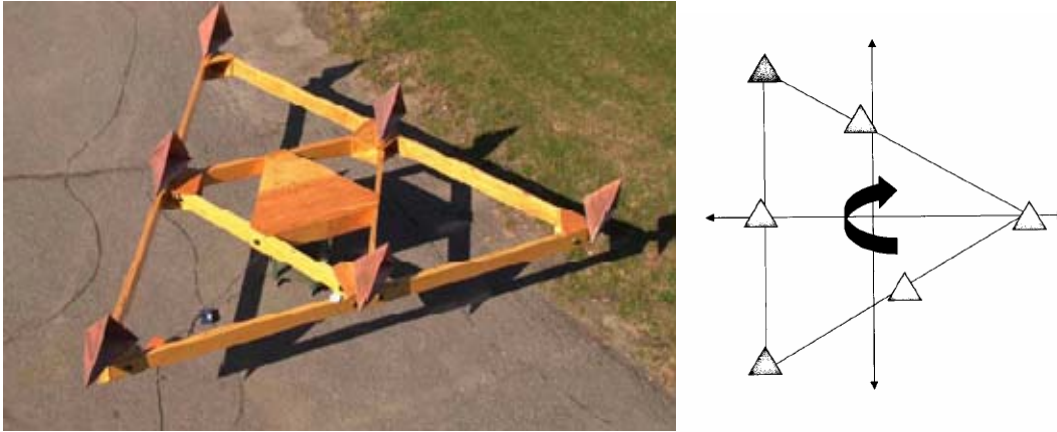


Figure 2. Picture of 6 – Point Delta Wing (From [1].)
Note that one reflector is offset to give a sense of orientation.

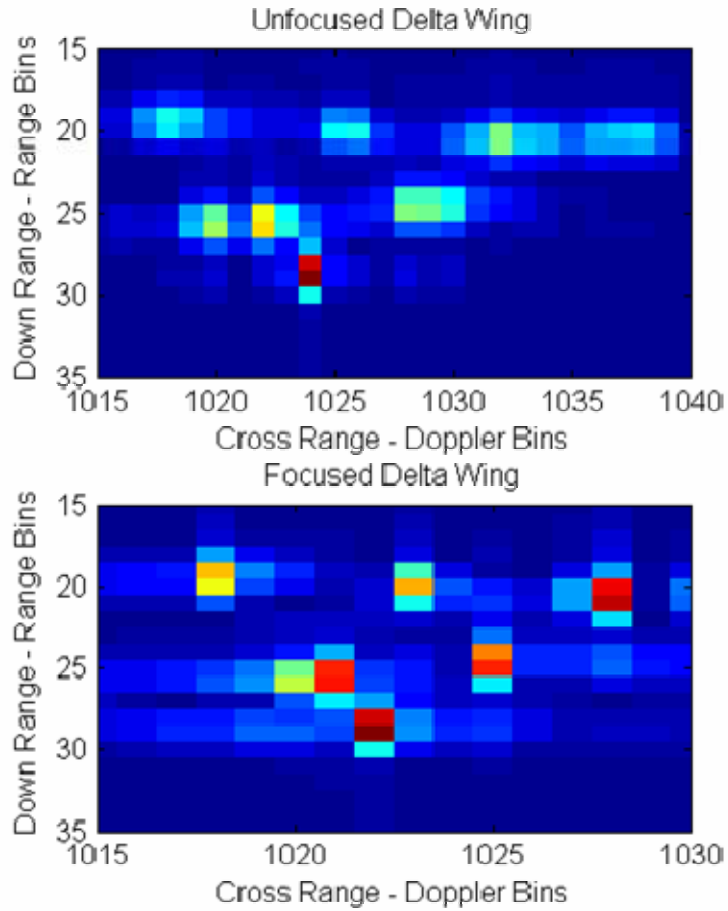


Figure 3. Example of an Unfocused and Focused 6 – Point Delta Wing.

These two data sets were used to test the algorithms designed in this thesis. The B727 simulated data set is an excellent data set since its ideal motion error characteristics

make it perfect for testing the first cut of an imaging algorithm. The 6 – Point Delta Wing data set is an experimental data set and it is particularly useful since the entire data set consists of 60,000 pulses in the cross-range. The 60,000 pulses can be cut into different size imaging intervals with each of these intervals displaying a different amount of error. Also the 6 – Point Delta Wing experimental data set is the only one of the two data sets that can be used for the 3D motion detection portion of the thesis.

2. Primary Research Questions

Based on the objectives above, there are two research questions that will be the primary subject of this thesis:

- a) Can the effective application of a genetic algorithm or a particle swarm optimization algorithm on the parameter search of the basis function for the AJTF algorithm significantly reduce the computation burden required to focus the ISAR image?
- b) Can the fast AJTF algorithm found in the first part of this thesis be applied to the 3D motion detection problem in order to rapidly sort the good imaging intervals, which obey the 2D motion assumption, from the imaging intervals that contain a high degree of 3D motion and cannot be focused?

3. Organization

This thesis is organized into six parts.

- i. Chapter II introduces ISAR signal processing to the extent required to understand the Adaptive Joint Time-Frequency Algorithm (AJTF);
- ii. Chapter III discusses time-frequency transforms and the AJTF algorithm, as found in [1], [2] and [3];
- iii. Chapter IV presents the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) Algorithm that were written to optimize the AJTF algorithm along with the ISAR imaging results;
- iv. Chapter V presents the algorithm for 3D motion detection and ISAR imaging results of imaging intervals determined to possess a high and low degree of 3D motion.

- v. Chapter VI summarizes the results that were achieved in this thesis and offers conclusions and recommendations for future work; and
- vi. The Appendix lists the MatLab code that was written for this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

II. INTRODUCTION TO INVERSE SYNTHETIC APERTURE RADAR (ISAR) PROCESSING AND THE FOCUSING PROBLEM

This chapter introduces the basic equations required to understand ISAR imaging and the focusing problem, which occurs when non-stationary signals such as time-varying Doppler are present in the ISAR return signal. It shows how Doppler resolution is related to the coherent processing interval (CPI) and how this relates to the amount of motion error in an image.

For the purposes of this thesis, the ISAR return signal is at the point in the processing where all of the point scatterers have been placed in their proper range bins and a fast Fourier transform (FFT) will focus the scatterer in their appropriate cross-range bins. The ISAR signal without error can be represented as [1,3]:

$$s(x, t) = \sum_{k=1}^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [R_0 + x_k \cos(\theta(t)) + y_k \sin(\theta(t))] \right\}, \quad (1)$$

where x is the range bin, N_k is the number of scatterers in the range bin x , A_k is the magnitude of the radar return signal, R_0 is the distance to the rotational center of the target which is constant during the coherent processing interval, (x_k, y_k) is the down-range and cross-range distance from the target's rotational center to the k^{th} point scatterer. Since no motion error is being assumed at this point, $\theta(t)$ can be written as [1], [3]:

$$\theta(t) = \theta_0 + \Omega t, \quad (2)$$

where Ω is the constant angular velocity of the radar target from which the Doppler cross-range location of the scatterer is extracted. If $\theta(t)$ is assumed to be small, Equation (1) reduces to [3]:

$$s(x, t) = \sum_{k=1}^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [R_0 + x_k + (\Omega t) y_k] \right\}, \quad (3)$$

which can easily be focused by the fast Fourier transform since the extracted Doppler is time-invariant. The time-invariance of the Doppler is critical for successful focusing using the fast Fourier transform.

The second equation of importance deals with the Doppler resolution, Δf_D , and its associated effect on the cross-range resolution, Δr_{cr} . Doppler resolution is associated with the coherent processing interval, T_{CPI} , such that [3]:

$$\Delta f_D = \frac{1}{T_{CPI}}, \quad (4)$$

which states that as the coherent process interval is increased, the Doppler resolution increases. Similarly, resolution in cross-range can be defined as [3]:

$$\Delta r_{cr} = \left(\frac{c}{2\Omega f_0} \right) \Delta f_D = \left(\frac{c}{2\Omega f_0} \right) \frac{1}{T_{CPI}}. \quad (5)$$

Therefore, from Equation (5), it is clear that, as the coherent processing interval is increased, the size of the Doppler bins in the cross-range becomes smaller and it is possible to resolve two different point scatterers in the same range bin even if they are only slightly separated in Doppler. In this thesis, the number of pulses in the cross-range is analogous to T_{CPI} . A greater T_{CPI} leads to a greater number of pulses in the cross-range and, subsequently, an ISAR signal with greater Doppler separation between point scatterers that are collocated in the same range bin.

If Equation (3) held for real ISAR returns from fast moving targets, it would make sense to use the maximum number of radar pulses (by maximizing the T_{CPI}) to form the high resolution image. However if the entire 60,000 pulses ($T_{CPI} = 30$ seconds) of the 6 – Point Delta Wing experimental data set is focused using the fast Fourier transform, the resulting image, as shown in Figure 4, is extremely blurred and bears no resemblance to the actual target in Figure 2. Even if a more reasonable T_{CPI} is chosen, such as a $T_{CPI} = 1$ second with 2048 pulses in the cross-range, the motion error is still significant enough to cause the image to blur beyond recognition. This can be seen by referring to the unfocused image in Figure 3.

If, instead, the T_{CPI} is reduced to 0.128 seconds so that there are now only 256 pulses in the cross-range, motion error can be avoided. The side effect of this choice is that the cross-range resolution is greatly reduced. Most imaging intervals will generate

images that resemble those found in Figure 5. These images possess no noticeable error in the cross-range but also possess no resolution in the cross-range with most point scatterers collapsed together on top of each other in the center Doppler bin.

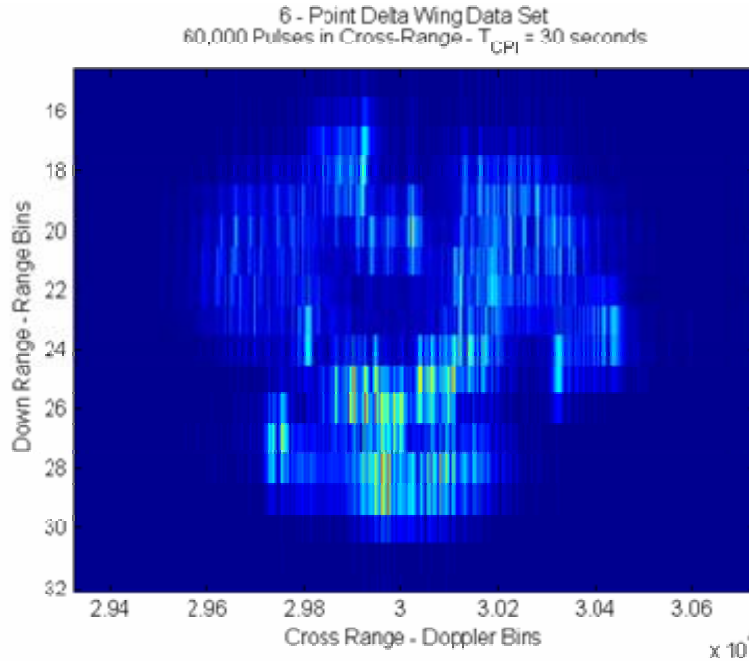


Figure 4. 60,000 Pulse 6 – Point Delta Wing Data Set Focused with the Fast Fourier Transform.

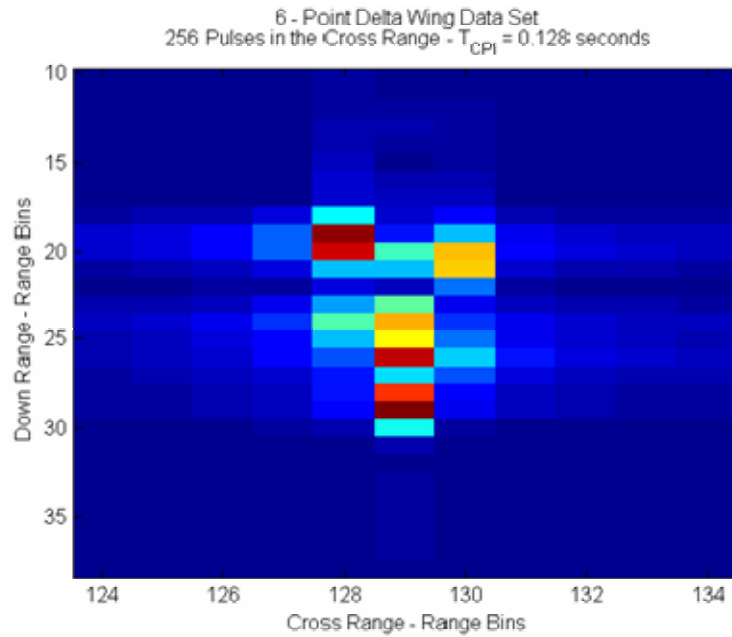


Figure 5. 256 Pulse Imaging Interval 6 – Point Delta Wing Data Set Focused with the Fast Fourier Transform.

Therefore, our motivation for a fast and effective motion compensation algorithm becomes obvious. Any imaging interval with a T_{CPI} that leads to good separation in Doppler of the point scatterers within the same range bin will possess excessive motion error, time-varying Doppler and a blurred image. An imaging interval with a small T_{CPI} will have no error, time-invariant Doppler but also no separation of point scatterers in the cross-range. What is required is an algorithm that can remove the time-varying Doppler from an imaging interval of sufficiently long T_{CPI} such that the resulting image is both well focused and well separated in Doppler.

With our requirement for an algorithm to correct motion error in an ISAR signal now clear, we can proceed to Chapter III where time-frequency transforms and the Adaptive Joint Time-Frequency algorithm are introduced. Time-frequency transforms and the AJTF are both crucial tools in achieving effective motion compensation in an ISAR signal.

III. INTRODUCTION TO TIME-FREQUENCY TRANSFORMS AND THE ADAPTIVE JOINT TIME-FREQUENCY ALGORITHM

As stated in Chapter II, in order to form good ISAR images that are well resolved in the cross-range, there is a requirement for a coherent processing interval, T_{CPI} , that is sufficiently long such that the cross-range resolution, Δr_{cr} , is acceptably small. Unfortunately a T_{CPI} that leads to a small Δr_{cr} also leads to blurring in the cross-range since the Doppler of the target's scatterers is no longer stationary, which is a critical requirement of the FFT. In this chapter, time-frequency transforms are introduced so that the non-stationary nature of the time-variant Doppler can be understood. Also, the AJTF algorithm, which is an effective way to compensate for time-variant Doppler is introduced.

A. TIME-FREQUENCY TRANSFORMS

As stated in [1], [3] and [5], the one weakness of the fast Fourier transform (FFT) is that it loses all information on how the signal changes with time. As explained in Chapter II, ISAR signal processing that uses the assumption of time-invariant Doppler during the coherent processing interval, T_{CPI} , will inevitably introduce severe blurring into the final high resolution image. Therefore, it becomes necessary to introduce the time-frequency transform and how it can be used to create an analogy between what is occurring in the time-frequency plane, the Fourier Spectrum and the radar image.

1. Short Time Fourier Transform

As discussed in [5], the short time Fourier transform (STFT) is a unique and straightforward way of introducing a time dependency into the Fourier transform. This is done by pre-windowing the FFT around a certain time instance and can be defined as [5]:

$$S_x(t, \nu, h) = \int_{-\infty}^{+\infty} s_x(u) h^*(u-t) e^{-j2\pi\nu u} du \quad (6)$$

where S_x is the STFT of range bin x , s_x is the ISAR signal for range bin x and $h^*(u-t)$ is the complex conjugate of the windowing function, which for purposes of this thesis is the default Hamming window. Figure 6 shows the time-domain waveform of an arbitrary

ISAR radar signal and the Hamming window. To generate the STFT, the Hamming window is moved across the radar signal and at each time instance the FFT is taken of the result of the multiplication of the window with the ISAR radar signal.

Using the time-frequency toolbox from [5], the STFT of an ISAR radar signal can be generated and the analogy between what is shown by the STFT, seen in the power spectrum of an ISAR range bin and displayed in the radar image, can be illustrated. Figure 7 shows these three graphs when taken from a blurred ISAR image. By examining range bin 31 in the close-up of the B727 blurred ISAR image (left), it can be seen that there appears to be two scatterers smeared across several Doppler bins in the cross-range. The same deduction can be made by examining the power spectrum (upper right), as it can be seen that there are two Doppler smeared point scatterers located in this range bin. Finally, the STFT (lower right) can be examined. From the STFT, it can be very clearly seen that there are two point scatterers located in range bin 31. The STFT also shows the nature of the time-varying Doppler of the two point scatterers as the ISAR signal is integrated from pulse to pulse during the coherent processing interval. The first point scatterer's Doppler decreases from pulse to pulse while the second point scatterer's Doppler increases from pulse to pulse.

Figure 8 shows the same information as Figure 7; however, it is for a focused ISAR image. Examining the close-up ISAR image from a focused B727 data set, it can be seen that the point scatterers in range bin 31 have collapsed in the cross-range and are well focused. It can also be seen that these scatterers have been shifted in Doppler. Since this shift is applied equally across the image, it has no adverse affect. More important are the results that are seen in both the STFT and the power spectrum of range bin 31. The STFT shows that the two point scatterers, which are now lines with zero slope in the time-frequency plane, no longer possessing time-varying Doppler during the coherent processing interval. This characteristic is echoed in the power spectra of the two point scatterers which now come to sharp points rather than exhibiting smearing of their power spectra across several Doppler bins as occurred in the unfocused image.

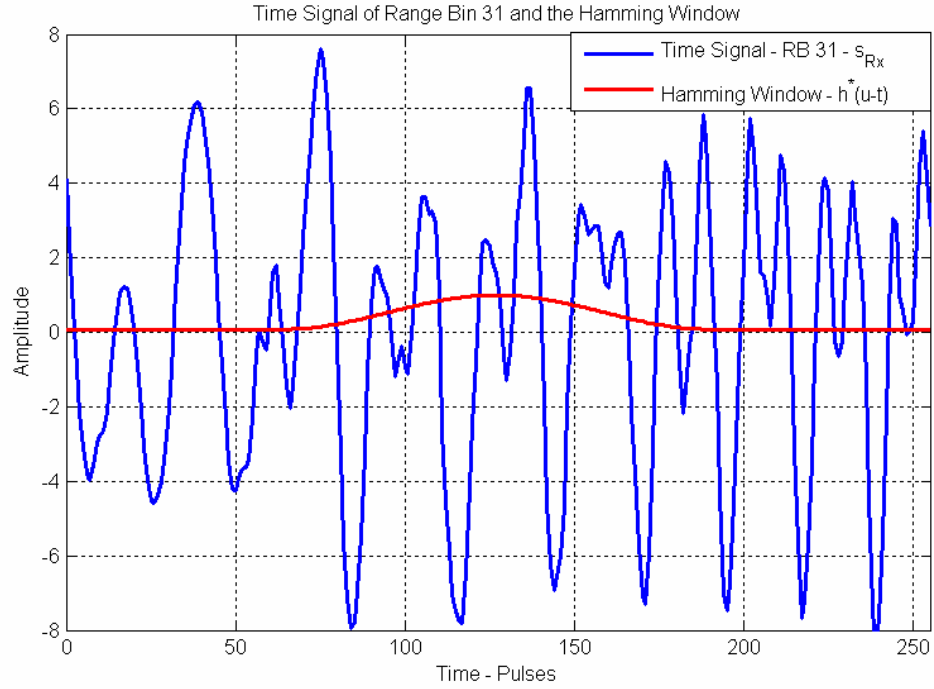


Figure 6. ISAR signal, s_{Rx} , of Range Bin 31 and the Hamming Window (After [5].)

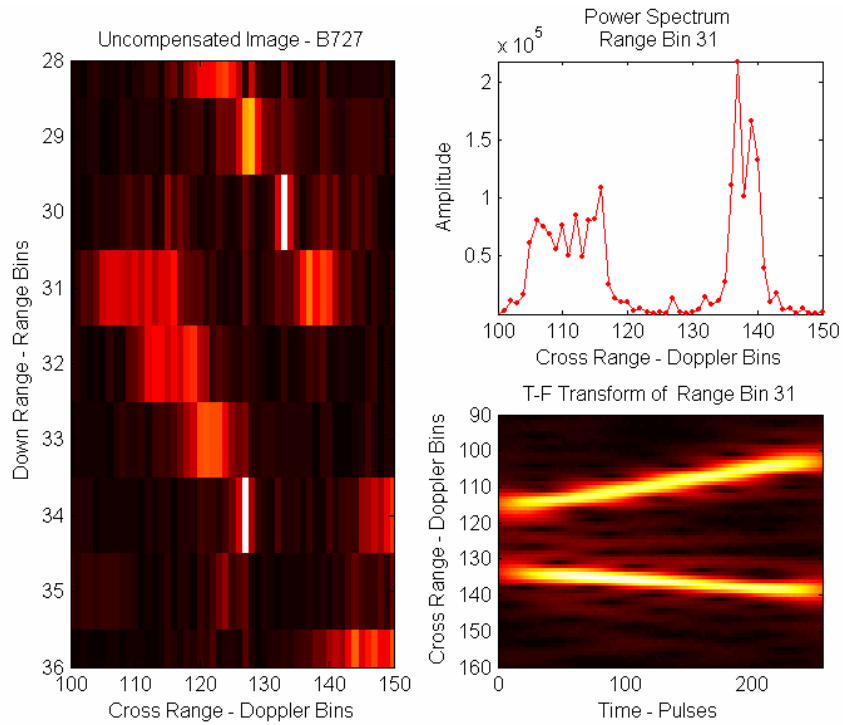


Figure 7. Blurred ISAR Image (close-up of Range Bin 31 in B727 data set), Power Spectrum of Range Bin 31 and its STFT.

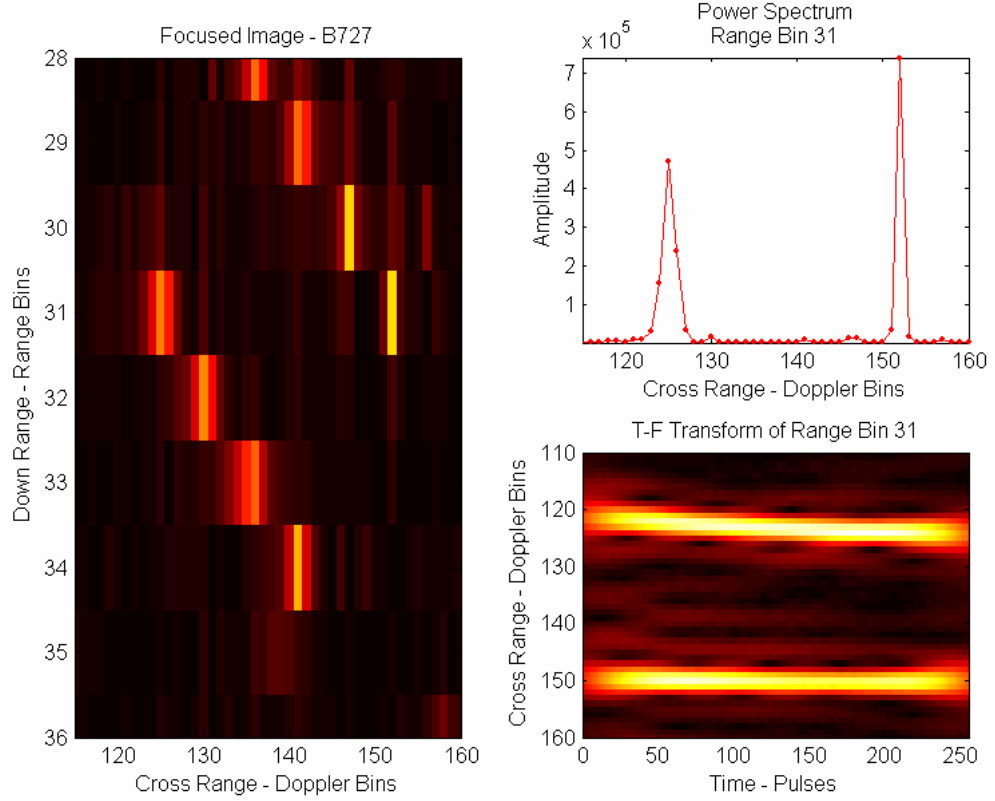


Figure 8. Focused ISAR Image (close-up of Range Bin 31 in B727 data set), Power Spectrum of Range Bin 31 and its STFT.

In concluding this section, it is important to reiterate the following points as they will become critical in chapter 4 when image focusing is discussed:

a) The range bins of a blurred ISAR image will have point scatterers that appear as lines with slopes (or even curved for higher order motion) when transformed to the time-frequency plane and the slope is an indication of the presence of time-varying Doppler. This same range bin will have the peaks of its power spectrum, which are produced by each point scatterer in the range bin, smeared across several Doppler bins when time-varying Doppler is present in the signal.

b) The range bins of a focused ISAR image will have point scatterers that appear as lines with a slope of zero in the time-frequency plane. The power spectrum of these same range bins will have sharp, well defined peaks located solely in their appropriate Doppler bins with minimum spillage into adjacent bins.

2. Weaknesses of the STFT and Time – Frequency Transform

Despite the usefulness of the STFT and other time – frequency transforms, as listed in [5] and [3], there are two primary weaknesses which will be of concern in the focusing problem of this thesis. The first weakness is the computational time of the STFT, which is the fastest time-frequency transform available in the time-frequency toolbox from [5]. Calculation time for the STFT is significantly longer than the FFT and this becomes critical if the STFT must be computed multiple times. The second weakness of the STFT, as stated in [2] and [5], is that it possesses poor resolution and cannot be simultaneously well resolved in both time and frequency. This is not a significant concern in the B727 simulated data set as all point scatterers in each range bin are well separated in Doppler. However, for the 6 – Point Delta Wing experimental data set, the point scatterers are not well separated in Doppler and it is difficult to see how the Doppler of each point scatterer changes with time. To increase time-frequency resolution, as recommended in [3] and [5], a different time-frequency transform could be chosen; however, this typically results in an increase in computational time, or in the case of the Wigner-Ville Distribution (WVD), cross-terms appear in the time-frequency image. As shown in Figure 9, which is a WVD of range bin 31 of the unfocused B727 simulated data set, these cross-terms can make the interpretation of the time-frequency image very difficult and the automation of line detection close to impossible.

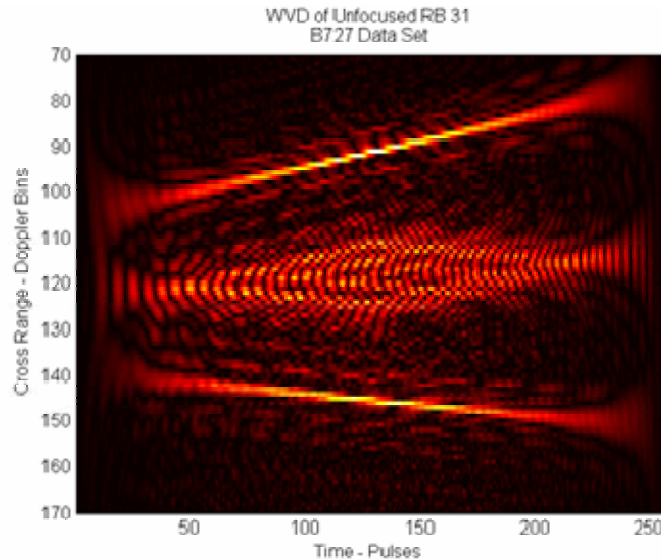


Figure 9. Wigner-Ville Distribution of Unfocused Range Bin 31 (After [3] and [5].)

B. ADAPTIVE JOINT TIME-FREQUENCY ALGORITHM

The AJTF algorithm can be found in [1], [2] and [3] and is capable of removing time-varying Doppler in a radar signal that is due to translational and rotational motion error. As explained in the references, the capability of the AJTF is limited by the holding of three primary assumptions. The first assumption is that the ISAR signal has been error corrected in the down-range direction and that each point scatterer is in its proper range bin. The second assumption is that of a rigid body, which implies that motion error that exists in one point scatterer will exist in all point scatterers of the target. This assumption is violated in the presence of jet engine modulation (JEM), helicopter rotors and the micro-Doppler phenomenon. The third assumption is that motion occurs in a 2D plane with respect to the radar. The AJTF algorithm is currently unable to correct for 3D motion. For this thesis, the first two assumptions will always hold. The assumption of 2D motion does not hold for all imaging intervals of the 6 – Point Delta Wing experimental data set and, as a result of the 3D motion error, these imaging intervals will not image well. As stated in the introduction, it is the second part of this thesis that deals with 3D motion error so, for now, it will be assumed that all three assumptions hold.

1. The Motion Model for a Moving ISAR Target

The motion model for an ISAR target is introduced in [2] and further explained in both [3] and [1]. The following explanation is drawn from all three sources but primarily [1] and [3]. Figure 10 shows the motion geometry of an ISAR target. Coordinates (U, V, W) is the radar coordinate system, while coordinates (x, y, z) is the coordinate system used to define the target and (X, Y, Z) is the coordinate system translated from the radar and is used to describe the target's rotation. It is centered at the geometric center of the target, about which the target will rotate. If $P(x, y)$ is the location of a point scatterer, then the initial distance from the radar to the k^{th} point scatterer located on the target at $t = 0$ is [3]:

$$R_k \cong R_0 + x_p \cos(\theta_0 - \alpha) + y_p \sin(\theta_0 - \alpha), \quad (7)$$

where R_0 is the initial line-of-sight distance from the radar to the target's geometric center, θ_0 is the initial angle between (X, Y, Z) and the (x, y, z) coordinate system and α is the azimuth angle of the scatterer off of the (U, V, W) axis.

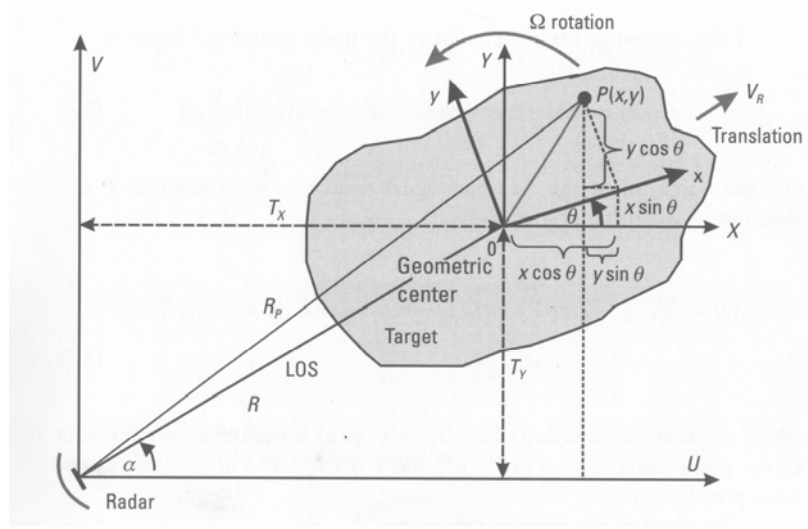


Figure 10. Geometry of the Radar Imaging of a Target (From [3].)

Now, if the target possesses a rotational motion, Ω , about the Z axis and a translational motion with radial velocity, V_R , then both the range and rotational angle become a function of time and Equation (7) become time dependent. The time dependent $R(t)$ can be expanded into a Taylor series polynomial as [1]:

$$R(t) = R_0 + V_R t + \frac{1}{2!} V_R' t^2 + \frac{1}{3!} V_R'' t^3 + \dots, \quad (8)$$

where $V_R' = \frac{\partial V_R}{\partial t} = a_R$ which is the acceleration of the scatterer in the radial direction and the additional derivatives of V_R are terms of higher order translational motion. The time dependent $\theta(t)$ can be expanded into a Taylor series polynomial as [2]:

$$\theta(t) = \theta_0 + \Omega t + \frac{1}{2!} \Omega' t^2 + \frac{1}{3!} \Omega'' t^3 + \dots, \quad (9)$$

where $\Omega' = \frac{\partial \Omega}{\partial t}$ which is the initial angular acceleration of the point scatter and the additional derivatives of Ω are terms of higher order rotational motion. Taking Equations (8) and (9), substituting them into Equation (7) and arbitrarily setting α and θ_0 to zero, we have [2]:

$$R_k(t) = R_0 + V_R t + \frac{1}{2!} V_R' t^2 + \dots + x_k \cos\left(\Omega t + \frac{1}{2!} \Omega' t^2 + \frac{1}{3!} \Omega'' t^3 + \dots\right) + y_k \sin\left(\Omega t + \frac{1}{2!} \Omega' t^2 + \frac{1}{3!} \Omega'' t^3 + \dots\right). \quad (10)$$

Now taking Equation (10) and calculating the Taylor series expansion of the sine and the cosine about t yields [1]:

$$R_k(t) = (R_0 + x_k) + (\Omega y_k + V_R) t + \frac{1}{2!} (V_R' - \Omega^2 x_k + \Omega' y_k) t^2 + \dots \quad (11)$$

which is the time-varying range to a one point scatterer on the target.

2. The ISAR Signal and the Motion Compensation Algorithm

Now Equation (11) can be substituted into Equation (1) which gives the equation for an ISAR return signal from a collection of N_k point scatterers located in range bin x where the ISAR return signal is not an ideal, errorless signal [1]:

$$s(x, t) = \sum_{k=1}^{N_k} A_k \exp\left\{-j \frac{4\pi f_0}{c} \left((R_0 + x_k) + (\Omega y_k + V_R) t + \frac{1}{2!} (V_R' - \Omega^2 x_k + \Omega' y_k) t^2 + \dots\right)\right\}. \quad (12)$$

If, as in [1], the following substitutions are made:

$$\begin{aligned} a_0 &= R_0 + x_k \\ a_1 &= \Omega y_k + V_R \\ a_2 &= \frac{1}{2!} (V_R' - \Omega^2 x_k + \Omega' y_k), \end{aligned} \quad (13)$$

the following observations are made. First, a_0 defines the coordinate position of the scatterer on the target. The second term, a_1 , is the radar line-of-sight translational motion and constant rotational motion and is linearly dependant on time. If all other terms are negligible, then a simple fast Fourier transform is sufficient to form a focused image.

The second term, a_2 (as well as subsequent terms if higher order motion is present), contains V'_R which is the translational motion error and $\Omega'y_k$ which is the rotational motion error that has a dependency on the cross-range. The $(\Omega t)^2$ can be neglected since [1] states that for centimeter and millimeter wave radars, this term is generally much less than 1.

Therefore, we can define the phases of the uncompensated range bin as the phase function [1]:

$$\Phi(t) = a_0 + a_1 t + a_2 t^2 + \dots \quad (14)$$

and since the phases in the range bin are time-varying, the instantaneous frequency can be expressed as [1]:

$$f_i(t) = \frac{1}{2\pi} \frac{\partial \Phi(t)}{\partial t} \quad (15)$$

which is also time-varying. Therefore, the Doppler frequencies of the point scatterers in the range bin are time-varying and the image is blurred in the cross-range.

The next step is to focus on a range bin and to find the translational motion compensation basis function of the form [3]:

$$\begin{aligned} h_{p_r}(t) &= \exp\{-j2\pi \cdot \Theta_T(t)\} \\ &= \exp\left\{-j2\pi \left(f_{11}t + \frac{1}{2!}f_{12}t^2 + \frac{1}{3!}f_{13}t^3 + \dots\right)\right\} \end{aligned} \quad (16)$$

where $\Theta_T(t)$ is the phase function for translational motion compensation and the parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ are the AJTF search parameters for translational motion compensation. A suitable basis function will closely resemble the prominent point scatterer in the time-frequency plane. Figure 11 shows an STFT of the unfocused range bin from the B727 simulated data set and the STFT of the best basis function for translational motion compensation (described later in this chapter). Once this basis function has been found, translational motion error can be corrected by applying the following equation from [1], [2] and [3]:

$$s(x, t)_T = s(x, t) \cdot \text{conj}\{h_{p_r}(t)\} \quad (17)$$

where $s(x, t)_T$ is the ISAR signal for all range bins with translational motion error removed, $s(x, t)$ is the ISAR signal for all radar bins, \cdot defines array multiplication and conj is the MatLab complex conjugate operator. Figure 12 shows the same range bin in the time-frequency plane after translational motion compensation. Note that the top line, representing the prominent scatterer in the range bin has had its slope zeroed which indicates the removal of time-varying Doppler. The remaining signal is now [1]:

$$s(x, t)_T = \sum_k^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [\Delta x_k + \Delta y_k \theta(t)] \right\} \quad (18)$$

where $\theta(t)$ is as defined in Equation (9) above.

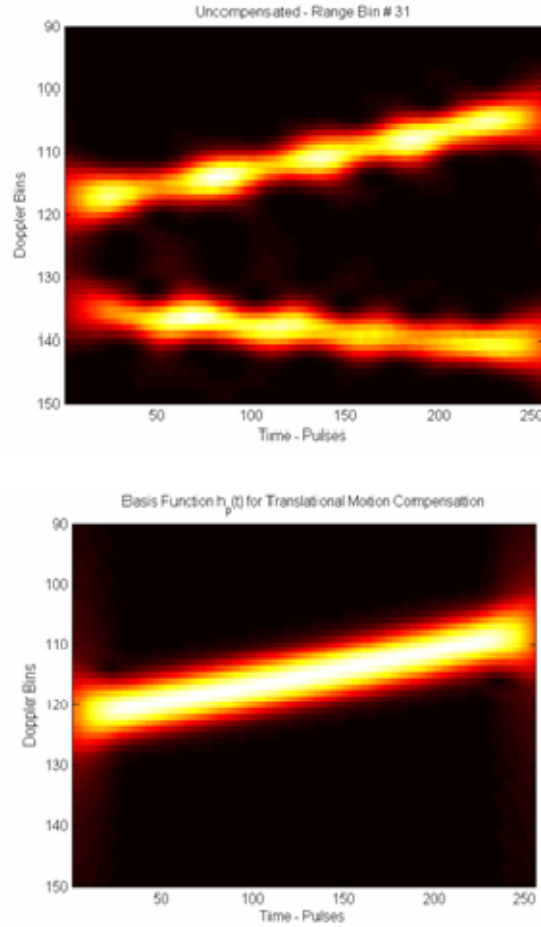


Figure 11. Unfocused Range Bin (top) and the Best Basis Function $h_{p_r}(t)$ for Translational Motion Compensation.

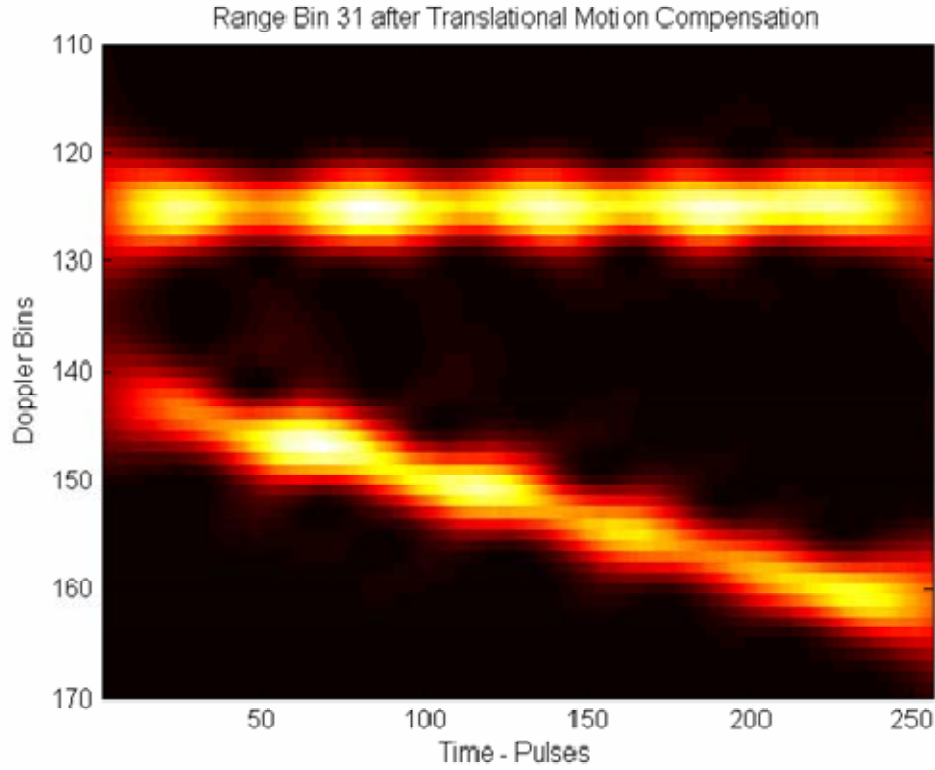


Figure 12. Time-Frequency Transform of Range Bin After Translational Motion Compensation.

Now it is required that rotational motion compensation be completed on the signal. This is done by choosing another range bin with a prominent point scatterer. If there is more than one scatterer in a range bin, it is possible to use the same range bin; however, the prominent point scatterer used for translational motion must first be removed from the signal. The algorithms written for this thesis used separate range bins for translational and rotational motion compensation. The first step is to again find a suitable basis function as in Equation (16) and as shown in Figure 9:

$$h_{p_R}(t) = \exp\{-j2\pi \cdot \Theta_R(t)\}. \quad (19)$$

However, for rotational motion compensation, it is the phases of the basis function in Equation (19), $\Theta_R(t)$, that are required [3]:

$$\Theta_R(t) = f_{21}t + \frac{1}{2!}f_{22}t^2 + \frac{1}{3!}f_{23}t^3 + \dots \quad (20)$$

where $\Theta_r(t)$ is the phase function for rotational motion compensation and the parameters $\{f_{21}, f_{22}, f_{23}, \dots\}$ are the search parameters for rotational motion compensation. The phase function of Equation (20) and the rotational angle in Equation (18) gives the relationship between rotational angle and dwell time such that [3]:

$$\theta(t) \propto \Theta_r(t). \quad (21)$$

The linear interpolation function in MatLab can reformat the ISAR data so that the radar signal is uniformly sampled in θ , only linearly dependent on time and the quadratic phase error is removed. The final focused ISAR signal, as given in [1], becomes:

$$s(x, t')_R = \sum_k^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [\Delta x_k + \Delta y_k \theta(t')] \right\} \quad (22)$$

where $s(x, t')_R$ is the ISAR signal after translational and rotational motion compensation, $\theta(t')$ is the uniformly sampled rotational motion term that only depends linearly on t' and t' is the new time variable.

3. Selection of an Appropriate Basis Function or Phase Function

In the last section, it was stated that suitable basis functions, $h_{p_r}(t)$ and $h_{p_r}(t)$, must be found in order to perform translational and rotational motion compensation. There are three methods that can be used to determine if a basis function is suitable for compensating for motion error. The most common method is the projection method which it is used in [1], [2], [3] and [6] as the way to determine the suitability of the basis functions. The second method follows the work in [16] and [17] by using the STFT and the Radon transform to determine basis function suitability by automatically recognizing the conditions of focus. The third method of determining basis function suitability uses the FFT to automatically recognize focus through the characteristics seen in the power spectrum. Note that for these three methods, first the best translational motion compensation parameters, $\{f_{11}, f_{12}, f_{13}, \dots\}$, are found and translational motion compensation is performed on the signal. After this has been completed, the rotational motion compensation parameters, $\{f_{21}, f_{22}, f_{23}, \dots\}$, can be found.

a. Projection Method to Determine $h_{p_T}(t)$ and $h_{p_R}(t)$

This approach is described mathematically in [1] as:

$$\{f_{11}, f_{12}, f_{13}, \dots\} = \arg \max \left| \left\langle s(x, t) h_{p_R}(t) \right\rangle \right| \quad (23)$$

where $\left| \left\langle s(x, t) h_{p_R}(t) \right\rangle \right|$ is the projection of the basis function onto the signal of range bin x in the time domain and can be defined, as in [1], as:

$$\left| \left\langle s(x, t) h_{p_R}(t) \right\rangle \right| = \int s(x, t) h_{p_R}^*(t) dt. \quad (24)$$

Rotation motion compensation parameters are found similarly as:

$$\{f_{21}, f_{22}, f_{23}, \dots\} = \arg \max \left| \left\langle s(x, t)_T h_{p_T}(t) \right\rangle \right| \quad (25)$$

where $\left| \left\langle s(x, t)_T h_{p_T}(t) \right\rangle \right|$ is defined in a similar fashion as Equation (24). Therefore, from the above equations, the most suitable basis function will be the one with the parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ or $\{f_{21}, f_{22}, f_{23}, \dots\}$ that, after vector multiplication between the signal and the basis function and after taking the magnitude of the result, will produce the greatest value. Though this method is a very popular method for determining basis function suitability, it was observed that the projection method did not always lead to an adequate basis function. To address this, the next two methods are discussed.

b. Automatic Recognition of Focus using the STFT and the Radon Transform

The Radon transform has been previously used in ISAR image formation in [16] and [17]. In this thesis, the Radon transform was used to determine the suitability of the basis and phase functions, $h_{p_T}(t)$ and $\Theta_R(t)$, and is a replacement for the previously explained projection method. Information on the calculation of the Radon transform is found in [13] and detection of lines in the time-frequency plane is introduced in [5]. This method finds its validity in the argument, as presented in Chapter III, that a focused scatterer which has had its motion error and, consequently, its time-varying Doppler removed, will be a line of zero slope in the time-frequency plane. The first step of this method is to take the current search parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ and formulate the ba-

sis function, $h_{p_r}(t)$. (If translational motion compensation has already been completed, the search parameters $\{f_{21}, f_{22}, f_{23}, \dots\}$ are used to form the phase function, $\Theta_R(t)$, for rotational motion compensation.) Once $h_{p_r}(t)$ or $\Theta_R(t)$ has been formulated, motion compensation, either translational or rotational as applicable, is performed on the range bin. It is only necessary to perform motion compensation on the range bin under examination during the basis and phase function searches since the rigid body assumption states that motion error is constant from range bin to range bin. The next step is to take the STFT of the resulting range bin and to calculate the Radon transform of the resulting image.

The Radon transform is a proven method for detecting lines in images (see [5] and [13]). It is able to return parameterized information on the lines found in the image. Figure 13 is the plot of the Radon transform of the STFT of range bin 31 in the B727 simulated data set in its unfocused state, after translational motion compensation and after rotational motion compensation. The images of the time-frequency transforms have already been displayed for these three states in Figures 11, 12 and 8, respectively. For the purposes of this method, only the θ parameter of the Radon transform is important as a value of $\theta = 90^\circ$ indicates a flat line in the subject time-frequency plane. Figure 13 demonstrates that, as lines in the time-frequency plane are flattened to zero slope when correct values of $\{f_{11}, f_{12}, f_{13}, \dots\}$ and $\{f_{21}, f_{22}, f_{23}, \dots\}$ are found and motion compensation is correctly applied, the peaks in the Radon transform graphs lie on the $\theta = 90^\circ$ line. The final Radon transform, which is taken from the STFT of a focused image shows that there are two lines in the image, one laying on the $\theta = 90^\circ$ line and the other on the $\theta = 89^\circ$ line. Therefore, detecting focus is simply a matter of searching for the parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ and $\{f_{21}, f_{22}, f_{23}, \dots\}$ that cause peaks along or very near the $\theta = 90^\circ$ line when motion compensation is applied.

The one significant drawback of this method which unfortunately makes it currently unusable, is that both the STFT and the Radon transforms take too long to calculate and even the most efficient of search algorithms must calculate these transforms many times. Therefore, this method cannot be currently used in ISAR image focusing in a real-time application. However, with the ever increasing computational power in to-

day's CPUs, it may one day be a suitable for determining basis function suitability. Fortunately, the research into this area led to the realization of a method that uses the FFT.

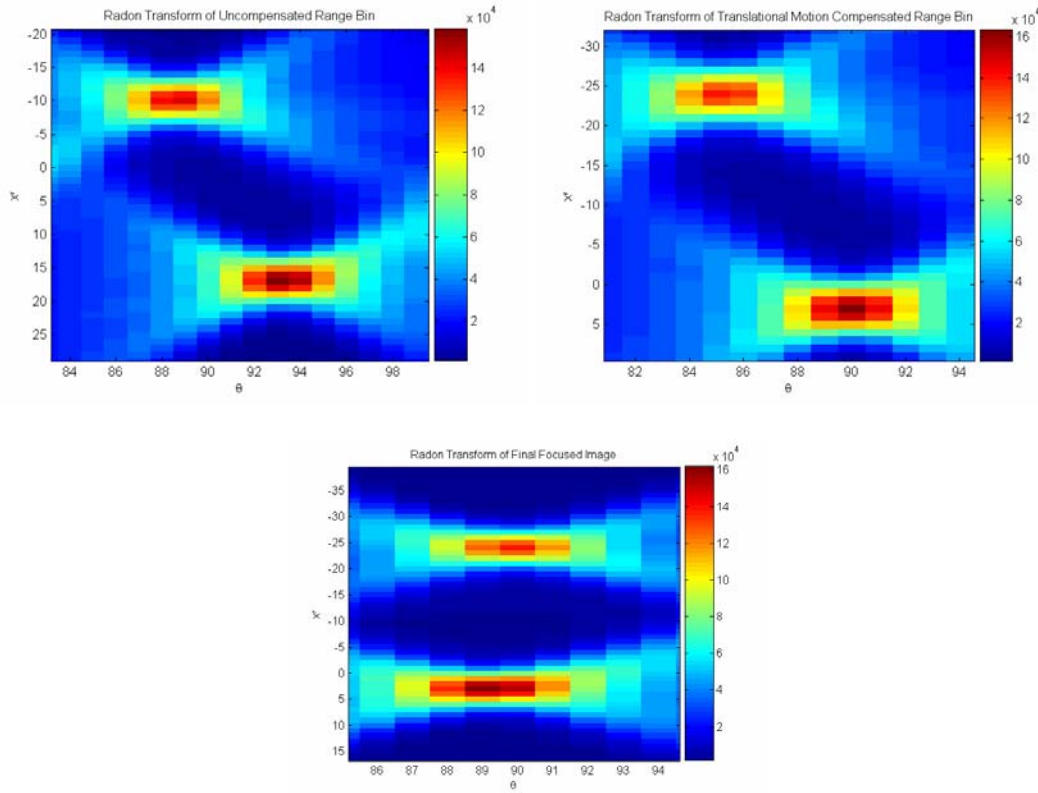


Figure 13. Radon Transform of Uncompensated and Compensated Range Bins.

c. *Automatic Recognition of Focus using the FFT*

This method was developed soon after it was realized that the method using the STFT and the Radon transform could not be practically implemented. The FFT method finds its validity in the explanation given in Chapter III.A in that a time-frequency transform of a range bin with motion error will have point scatterers mapped out as tilted (or possibly curved) lines and a power spectrum which has areas of maximum value smeared across several Doppler bins. A focused range bin will have a scatterers mapped out as lines with zero slope in the time-frequency plane and a power spectrum where maximums are sharp points in their respective Doppler (i.e., cross-range) bins (refer to Figures 7 and 8 for images that illustrate this).

The FFT method starts the same as the previous method in Subsection 3.B.b of this Chapter, in that the range bin under examination is subjected either to trans-

lational or rotational compensation using the test parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ or $\{f_{21}, f_{22}, f_{23}, \dots\}$, as applicable. The power spectrum of the range bin x is then calculated using the familiar equations:

$$S(f)_T|_x = \left| \text{FFT} \left(s(t)_T|_x \right) \right|^2 \quad (26)$$

and

$$S(f)_R|_y = \left| \text{FFT} \left(s(t)_R|_y \right) \right|^2 \quad (27)$$

where $s(t)_T|_x$ is the ISAR signal of range bin x after translational motion compensation and $S(f)_T|_x$ is its power spectrum, $s(t)_R|_y$ is the ISAR signal of range bin y after rotational motion compensation and $S(f)_R|_y$ is its power spectrum. Now to determine the suitability of the basis function, $h_{pr}(t)$, or the phase function, $\Theta_R(t)$, it is necessary to determine the following:

$$\{f_{11}, f_{12}, f_{13}, \dots\} = \arg \max \left\{ \frac{\max \{S(f)_T|_x\}}{\sum_x S(f)_T|_x} \right\} \quad (28)$$

and

$$\{f_{21}, f_{22}, f_{23}, \dots\} = \arg \max \left\{ \frac{\max \{S(f)_R|_y\}}{\sum_y S(f)_R|_y} \right\}. \quad (29)$$

Equations (28) and (29) state that the best search parameters would be ones that place the maximum percentage of the power spectrum found in the entire range bin in a single Doppler bin. A simple maximum search will not work since incorrect values of the search parameters can generate a high peak in a Doppler bin which is much greater than what will be found if the image were focused, but there will also be a very significant amount of the power spectrum in adjacent bins and accordingly the image will be extremely blurred. Taking the average is very effective at avoiding this situation because the averaging process will score these sets of parameters very low. An example of a

range bin, motion compensated with two different sets of parameters, one that does not focus the image and the other that focuses the image, is shown in Figure 14. This automatic recognition of focus technique using the FFT is very fast and is the technique that is used in the search algorithms in Chapter IV for determining the suitability of a basis function's or phase function's search parameters.

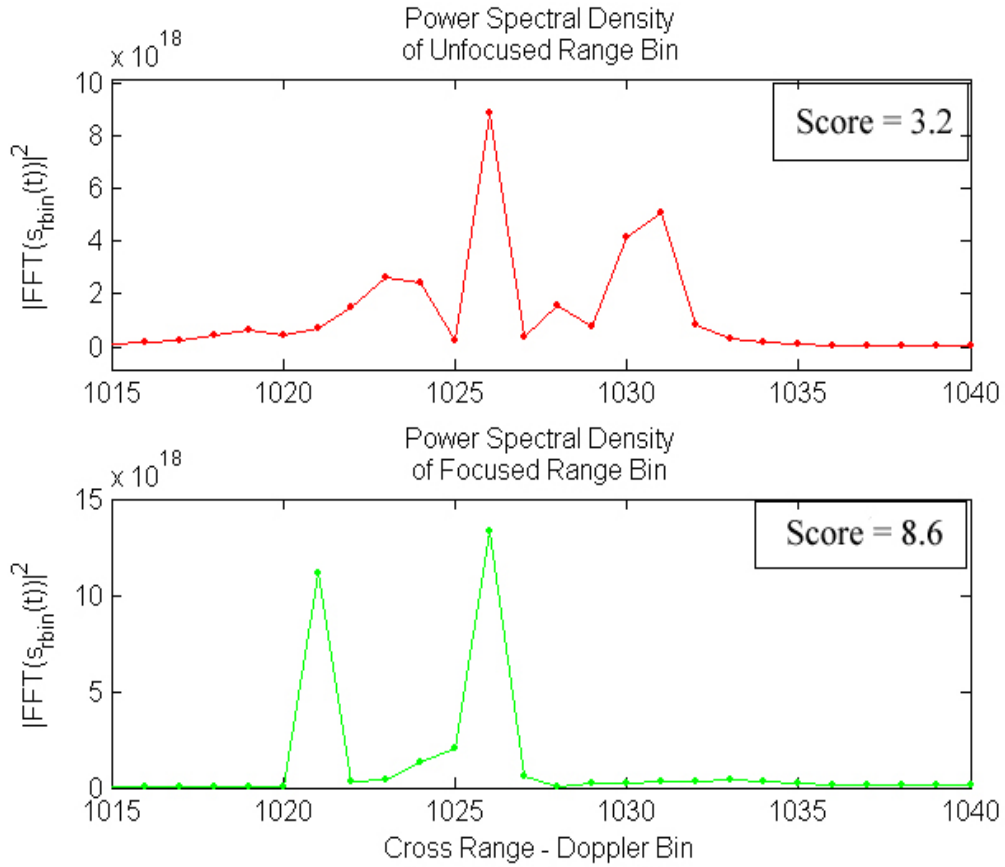


Figure 14. Example of a Unfocused and Focused Range Bin and Their Score.

C. CHAPTER SUMMARY

In this chapter, two key concepts, which are critical to Chapter IV and the overall problem of ISAR focusing, were introduced. First, the mathematical model of the AJTF algorithm was presented and it is this algorithm that enables time-varying Doppler due to target motion to be removed from the ISAR signal. Secondly, the FFT method of determining basis and phase function suitability was presented. Chapter IV uses this method as the core of the GA and the PSO search algorithms.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. GENETIC ALGORITHM AND PARTICLE SWARM OPTIMIZATION AS SEARCH ALGORITHMS

This chapter covers the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) algorithm and their application to extracting translational and rotational motion correction parameters from the solution space. The chapter concludes by offering a comparison between the two algorithms.

A. INTRODUCTION TO THE SOLUTION SPACE AND THE EXHAUSTIVE SEARCH

Before beginning the discussion on the GA and the PSO algorithms it is important to briefly discuss the construct of the solution space which must be searched and the exhaustive search method, as described in [1] and [3], that is typically used to find the search parameters $\{f_{11}, f_{12}, f_{13}, \dots\}$ or $\{f_{21}, f_{22}, f_{23}, \dots\}$. In the AJTF algorithm for ISAR image focusing, the parameters f_{11} and f_{21} are always found using the FFT such that:

$$f_{11} = \text{DopplerBin} \left\{ \arg \max \left(\left| \text{FFT} \left(s(t)|_x \right) \right|^2 \right) \right\} \quad (30)$$

and

$$f_{21} = \text{DopplerBin} \left\{ \arg \max \left(\left| \text{FFT} \left(s(t)_T|_y \right) \right|^2 \right) \right\} \quad (31)$$

which means that f_{11} and f_{21} are always equated to the Doppler bin possessing the maximum power spectral density in range bins x and y . The remaining parameters, $\{f_{12}, f_{13}, \dots\}$ and $\{f_{22}, f_{23}, \dots\}$, must be searched and each parameter can possess any real valued number lying between $-N$ to $+N$, where N is the number of pulses used to form the ISAR image in cross-range. Therefore, as the number of pulses that are used to form an image increases, so does the solution space and the computational intensity required to achieve focus.

Figures 15 and 16 shows a map of the solution space that must be searched for the B727 simulated data set when the search is limited to a two-dimensional search space (i.e., 3rd-degree motion error which requires a search for parameters $\{f_{12}, f_{22}\}$ and $\{f_{22}, f_{23}\}$). For translational motion compensation, it can be seen that the solution space

is fractured, with many areas of separate local maximums. Because of this fracturing, conventional searches, such as a gradient search, may have problems finding the global maximum and instead converge to a local maximum [9,14]. This local maximum may not be an acceptable set of parameters for focusing the image. The rotational motion solution space is not nearly as fractured but it does have a couple of valleys which may cause a gradient search to mistakenly converge to a local maximum.

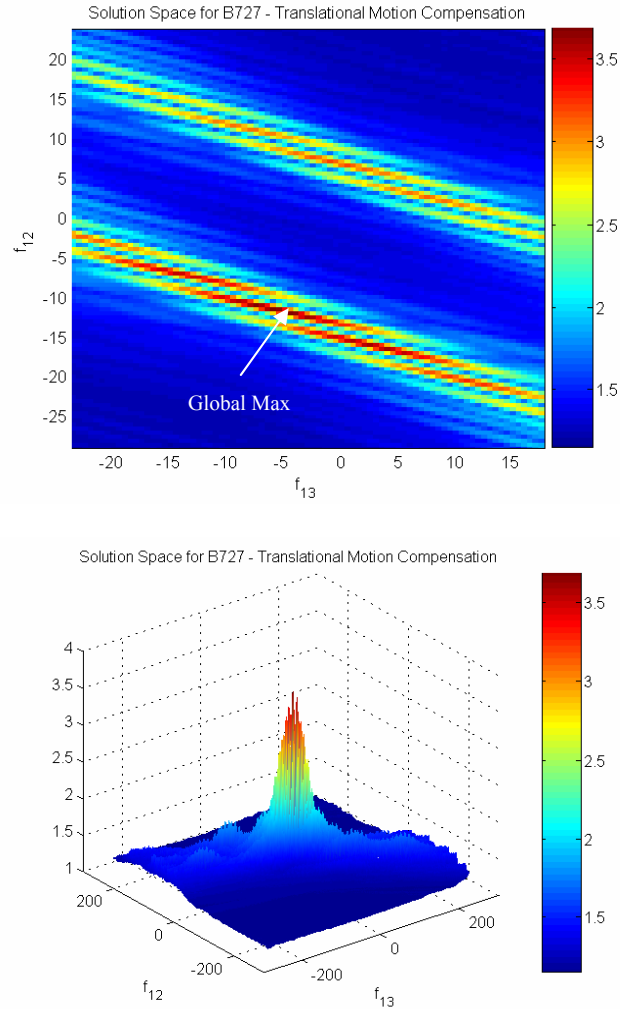


Figure 15. 2D and 3D Visualization of B727 Solution Space for Translational Motion Compensation.

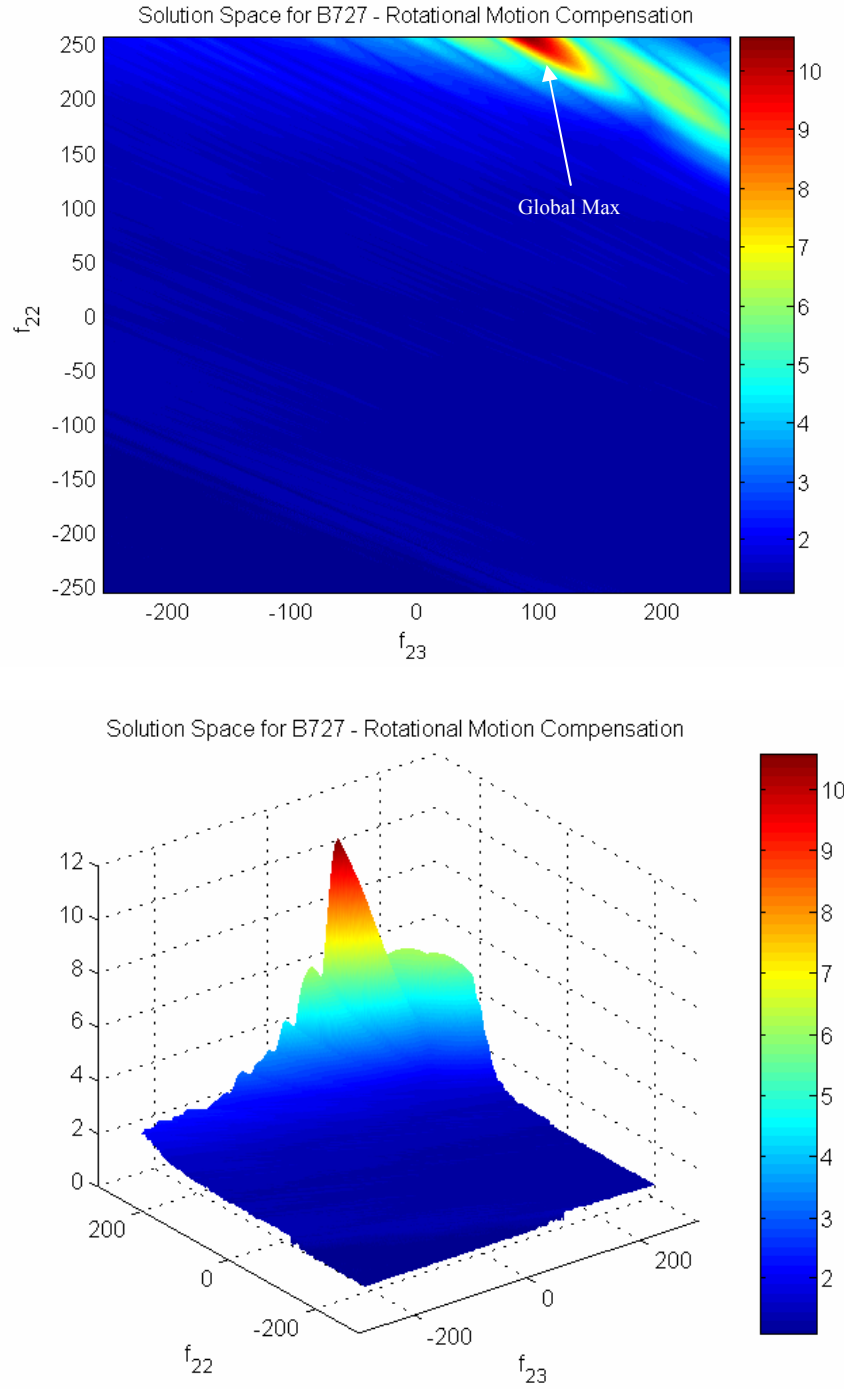


Figure 16. 2D and 3D Visualization of B727 Solution Space for Rotational Motion Compensation.

A common method for searching this solution space, as detailed in [1], [2] and [3], is the exhaustive search. The exhaustive search is typically done by finding f_{11} as in

Equation (30) and then stepping through f_{12} for the span of the solution space (while the additional parameters, f_{13}, f_{14} and higher terms are set to zero) and finding the best term that satisfies Equation (28). Once this term is found, the next term is stepped through the solution space, with previous terms set to their optimal values, until again the maximum of Equation (28) is found. This is repeated, as required, for all of the search parameters until the best basis function, $h_{p_r}(t)$, is found and translational motion compensation is performed on the ISAR signal. This identical methodology is then repeated to find the best phase function, $\Theta_R(t)$.

Though this is an effective method of finding the parameters of motion compensation, it suffers from two drawbacks. The first is that when searching for the third-order parameters (i.e., f_{13} or f_{23} and above), the exhaustive search may not find a true global maximum since it is doing a 1D search of a parameter with all other parameters fixed, despite the fact that the parameters are not orthogonal. Secondly, even with this reduced search space, it is still very computationally intensive and time consuming to step through all possible answers. One way of measuring computational intensity is to determine the number of times the cost function must be calculated. In terms of this exhaustive search algorithm, the cost function is Equation (28) and for the exhaustive search it must be calculated $N_{\text{cost_function}}$ times as determined:

$$N_{\text{cost_function}} = (2 * N_{\text{cross_range}}) \cdot N_{\langle f_{12}, f_{13}, \dots \rangle} \cdot N_{\langle f_{22}, f_{23}, \dots \rangle}, \quad (32)$$

where $N_{\text{cross_range}}$ is the number of pulses in the cross-range, $N_{\langle f_{12}, f_{13}, \dots \rangle}$ is the number of search parameters for translational motion compensation and $N_{\langle f_{22}, f_{23}, \dots \rangle}$ is the number of search parameters for rotational motion compensation. Therefore, for the B727 simulated data set with 256 pulses in the cross-range and 3rd-degree motion compensation (required to find two search parameters for both translational and rotational motion compensation), the number of cost function calculations is:

$$N_{\text{cost_function}} = (2 \cdot 256) \cdot 2 \cdot 2 = 2048. \quad (33)$$

Therefore, when the efficiency of the GA and PSO algorithm is measured, it will be evaluated not only by its run time but also by the number of cost function calculations it requires to converge. Minimizing the number of cost function calculations is the key to an efficient algorithm and a clear measure of how much better the algorithm is than the simple exhaustive search.

B. GENETIC ALGORITHM PARAMETER SEARCH

Genetic algorithms, first developed in 1989 by Goldberg, whose work is published in [9], are particularly well suited to searching a large solution space that contains one global maximum buried amid many local minima and which may be discontinuous. A GA search attempts to model natural behavior where strong traits will propagate through a breeding population while weak traits will die off and eventually be removed from a population after a number of generations. One outstanding feature of a well-written GA is that it will rapidly converge very close to the global maximum. The disadvantage is that once it has converged to this point, typically only small improvements will be made to the found global maximum and this only after many generations. This trait is illustrated in Figure 17 and is a typical feature of any evolutionary search. The GA experiences a rapid increase in the best score of its found global maximum in the first 23 generations of the evolutionary search. The next 32 generations are characterized by significant but smaller increases to the score of the found global maximum. The remaining generations are characterized by only a minimal and hardly noticeable increase to the score of the found global maximum. Therefore, using this argument, it is obvious to state that with GAs, convergence to the absolute and unique maximum is not guaranteed. However, the probability that convergence has occurred increases with the user's willingness to wait for the algorithm to process successive generations. Looking again at Figure 17, it can be stated with confidence that after the 70th generation, the GA has converged either at or near the true global maximum.

In ISAR imaging, there is no knowledge of the location of the true global maximum nor is there any way to predict the likeliness of finding a better global maximum. As successive generations occur, the probability of finding a better global maximum diminishes. Also, since the goal is to rapidly find acceptable search parameters that will focus the ISAR image, exiting the search as expediently as possible becomes critical.

Therefore, the rapid convergence (i.e., increase in the score of the found global maximum) of the GA early on in the search can be exploited. Combining this rapid convergence with the ability to optimally decide when to exit the search will allow for rapid focusing of ISAR images. The work of Li and Ling in [6] is an excellent first publication of a GA applied to ISAR focusing and was used as a reference in the writing of the GA code for this thesis. In particular, [6] states that real-valued GAs significantly outperform binary coded GAs. Because of this, the GA for this thesis will forego exploring binary-coded GAs and investigate only a real-valued GA. Another reference source was [10], which provides a good explanation of the flow and construct of a GA.

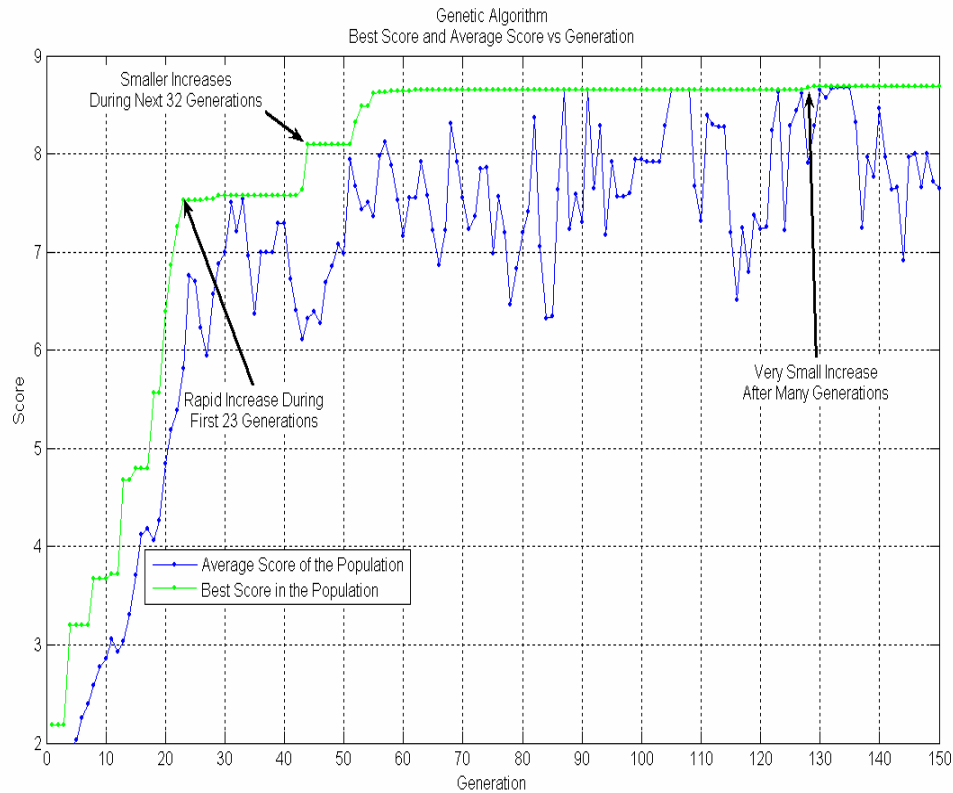


Figure 17. Illustration of a GA's Rapid Convergence.

1. Sequence of the Genetic Algorithm

The genetic algorithm follows a set sequence that repeats until done (see Figure 18) with each cycle referred to as a generation. The definition of each step is as follows:

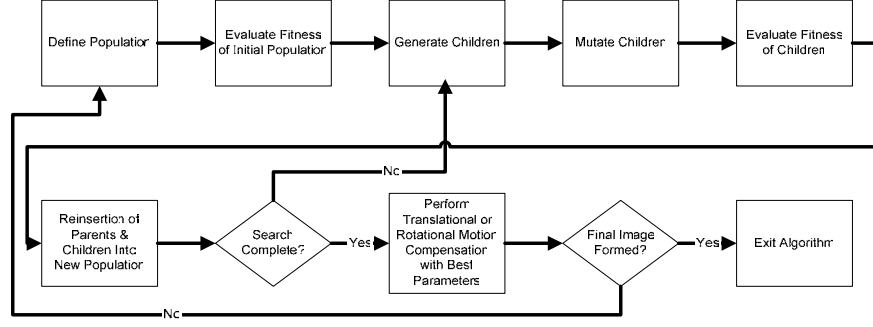


Figure 18. Flow Chart of the ISAR Genetic Algorithm (After [6].)

a. *Define the Population*

The first step of the GA is to generate a population of suitable size to ensure sufficient randomness in the initial group of parents. It is important, however, to ensure that the population is not so large that it becomes unwieldy and slows down the algorithm or that, after only a few generations of the GA, the cost function has been calculated more times than it would have been if an exhaustive search had been used. However, in the ISAR case, the ideal population size will be function of the number of pulses used in the cross-range, the number of search parameters due to the order of motion error and the maximum number of generations that the GA will run. For the ISAR parameter search, the initial population is a matrix with the first element of each parent being f_1 as determined by the FFT. (Note that from this point on when search parameters are referred to with a single subscript, the argument applies equally to translational and rotational motion compensation search parameters. The double subscript will be added when necessary to differentiate between the two sets of search parameters.) This parameter does not change. The remaining parameters, $\{f_2, f_3, \dots\}$, are initialized as uniform random variables between $\pm N$, where N is the number of pulses in the cross-range. After initialization, the population will resemble the construct shown in Table 1.

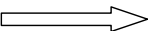


Degree				
P O P U L A T I O N S I Z E 	f_1	f_{2_1}	$f_{3_1} \dots$	 Parent 1
	f_1	f_{2_2}	$f_{3_2} \dots$	
	f_1	f_{2_3}	$f_{3_3} \dots$	
	f_1	f_{2_4}	$f_{3_4} \dots$	
	.	.	.	
	.	.	.	
	.	.	.	

Table 1. The Genetic Algorithm Population – 2nd subscript indicated Parent.

b. Evaluating Fitness of Each Member of Population

In any search algorithm, the ability to evaluate the suitability of a set of parameters compared to other sets of parameters when applied to the cost function is absolutely crucial in order to ensure that the best parameter set is chosen. In a GA, it is critical, since the fitness ranking will determine which parents will be used to form the next generation and which of the newly formed children will be inserted into the new population. Evaluating fitness is the key component in determining the search pattern of the solution space. Since this GA uses the FFT version of the automatic recognition of focus method to determine parameter fitness, all of the parents or children are evaluated against the cost function in Equation (28). In addition, because the calculated fitness between superior and poor parameters are small, a scaling exponential (chosen from empirical observation to be 100) was added to the fitness calculation in order to increase the separation between the fitness score of parameters. Care must be taken when choosing the exponential. Too low and it will make no difference to the efficiency of the search but set it too high and the GA will exit part of the solution space and may fail to find the global maximum that could reside there.

c. *Generating Children*

Now that the fitness of each parent has been computed, the next step is to form the children that will be part of the next generation. The first step is to determine, based on the fitness of each parent, the likeliness that a parent will be mated with another parent to form two children. This is done using the roulette wheel method as introduced in [9] and [10]. The way the roulette wheel works is that each parent is assigned a portion of the wheel that is proportional to the percentage they contribute to the sum of the fitness values of the entire population. This way parents who contribute significantly to the population have a significant chance of being chosen to produce children while even a poor contributor retains a chance, albeit a small one, to contribute to the population. It is critical that even poor parents be given a chance to reproduce since it ensures genetic diversity within the population and leads to a complete search of the solution space. The roulette wheel, shown in Figure 19, was built in MatLab by assigning a vector a cumulative sum of the fitness values (i.e., first value equal 3, second value equal to 13.5 and the last equal to 30.02).

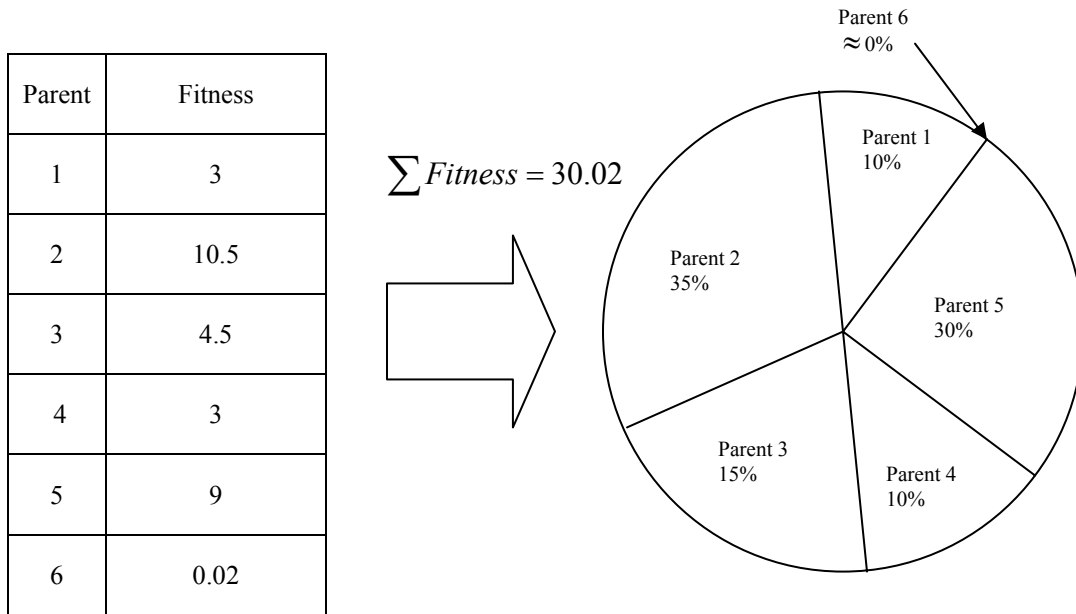


Figure 19. Illustration of the Roulette Wheel.

Once the roulette wheel has been constructed, the mating pool can be constructed. As in Figure 19, if our population is of size 6, we need 6 parents to form the mating pool. Choosing is done by generating a pointer which is computed as a linear distributed random variable between 0 and $\sum Fitness$. A parent is chosen if it is the first parent with a roulette wheel value greater than the random number assigned to the pointer. Figure 20 shows a pointer which possesses a value of 19. On the roulette wheel, Parent 3 has a value of 18 while Parent 4 has a value of 21. Since $19 > 18$ and $19 < 21$, Parent 4 is selected for the mating pool. Parent selection for the mating pool continues in this manner until all 6 slots in the mating pool are filled.

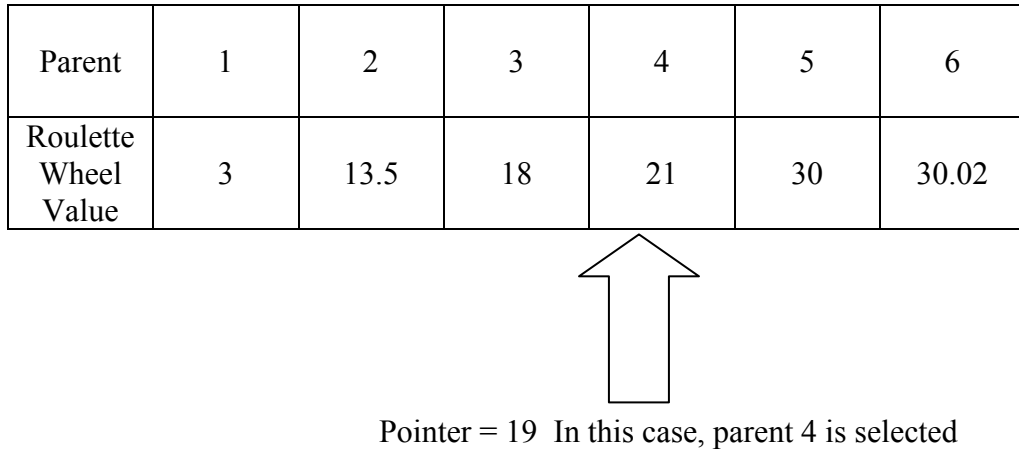


Figure 20. Illustration of the Mating Pool Selection Method.

Once the mating pool has been populated it is broken into two equal arrays and their order is randomized. The matching elements of the two breeding pools are the mating pairs. Breeding between the two mating pairs are determined by the cross-over equations taken from [6]:

$$C_1 = \alpha P_{1,1} + (1 - \alpha) P_{1,2} \quad (34)$$

and

$$C_2 = (1 - \alpha) P_{1,1} + \alpha P_{1,2} \quad (35)$$

where $P_{1,1}$ and $P_{1,2}$ are the first parents in the two breeding pools, C_1 and C_2 are their children and α is a uniformly distributed random variable between 0 and 1. So for each search parameter, the child is getting a new parameter that is equivalent to $f_{2_{CHILD1}} = (\alpha)f_{2_{PARENT1}} + (1 - \alpha)f_{2_{PARENT2}}$. Also remember that f_1 does not change as it is determined by the FFT.

d. Mutation of the Children

The purpose of mutation is to allow the search to be global in nature even as the parents converge towards each other and a possible global maximum. The mutation process occurs right after the children have been formed and is illustrated in Figure 21. Mutations are determined by [6]:

$$C_1 = \alpha C_1 + (1 - \alpha)P_{random}. \quad (36)$$

In Equation (36), C_1 is the child to be mutated, P_{random} is a parent pulled randomly from the solution space and α , as before, is a uniform random variable distributed between 0 and 1.

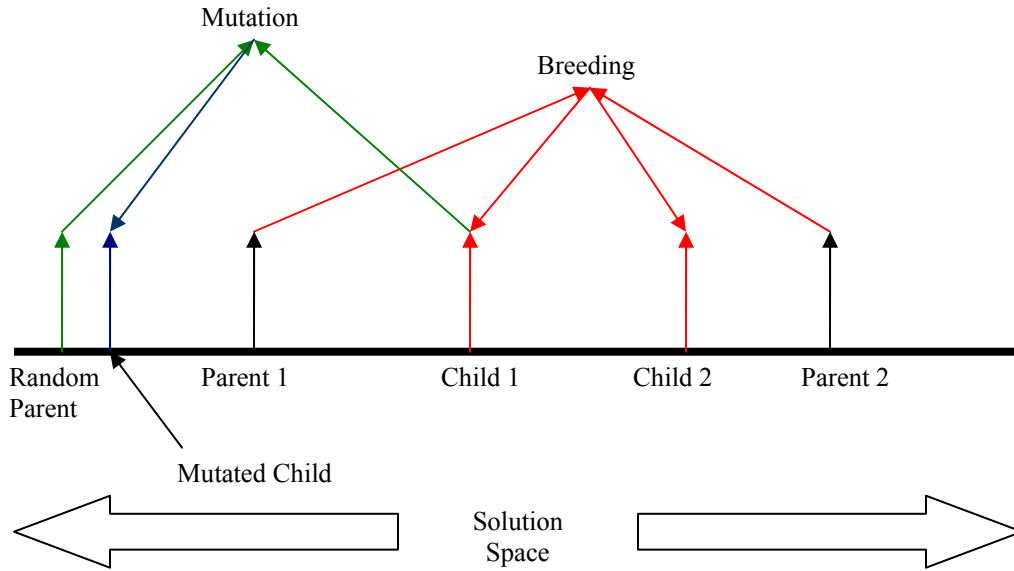


Figure 21. Illustration of the Mutation Process.

The other parameter that needs to be addressed is the mutation rate. Setting the rate too high destroys the integrity of the local search of a maximum and the GA may be terminated before it is able to find the absolute maximum. However, setting it too low increases the probability that the GA will converge to a local maximum. Reference [10] recommends a low mutation rate of 0.1% - 1% since, in nature, mutations are relatively rare. Reference [6] uses a mutation rate of 30% for their ISAR GA. Reference [11] suggests a variable rate, one which starts high and reduces during the number of cycles. Empirically, it was determined that the mutation rate from [6], set at a constant 30%, produced the best results. In this thesis' GA, this mutation rate was required to reliably break the GA out of a situation where it would repeatedly converged to a local maximum that created a poorly focused image.

e. Evaluating the Fitness of Each Child and Reinsertion of Parents and Children into New Population

Once the mutation process has been completed, each child is evaluated for its fitness as described above. The next stage is to devise a way to reinsert the old parents and the new children into the new population which promotes convergence to the global maximum. Reference [10] suggests that reinsertion should be a random process where only a certain percentage of the children are randomly chosen to join the new population and the remaining empty slots are randomly filled by parents of the old population. Care should be taken when applying this approach, particularly when speed of convergence is of particular importance. What occurs in this approach is that the current maximum (i.e., the best value found so far in the search) could depart the population, poor performers will enter into the population and high ranked children may not enter the population. A compromise to this, which was initially implemented in this thesis' GA, was setting the reinsertion rate of the children to an 80% probability of insertion into the new population with the remaining empty slots in new population being filled by the highest ranked parents. The probability of convergence graph, as a function of size of population and the number of generations evolved using this reinsertion method, was generated and is displayed in Figure 22. (Discussion on how the convergence graph is generated will be discussed later in the results section.) As can be seen from this graph, the probability of convergence, for 30 trials at each point, has a disappointing maximum value of 80% even with a large number of generations and a large population size. To improve this, it is

tempting to choose a deterministic approach where the best parents and children are chosen to form the new population. However, while building the GA for this thesis, implementing reinsertion in this manner tended to lead to convergence to a local maximum and not the global maximum.

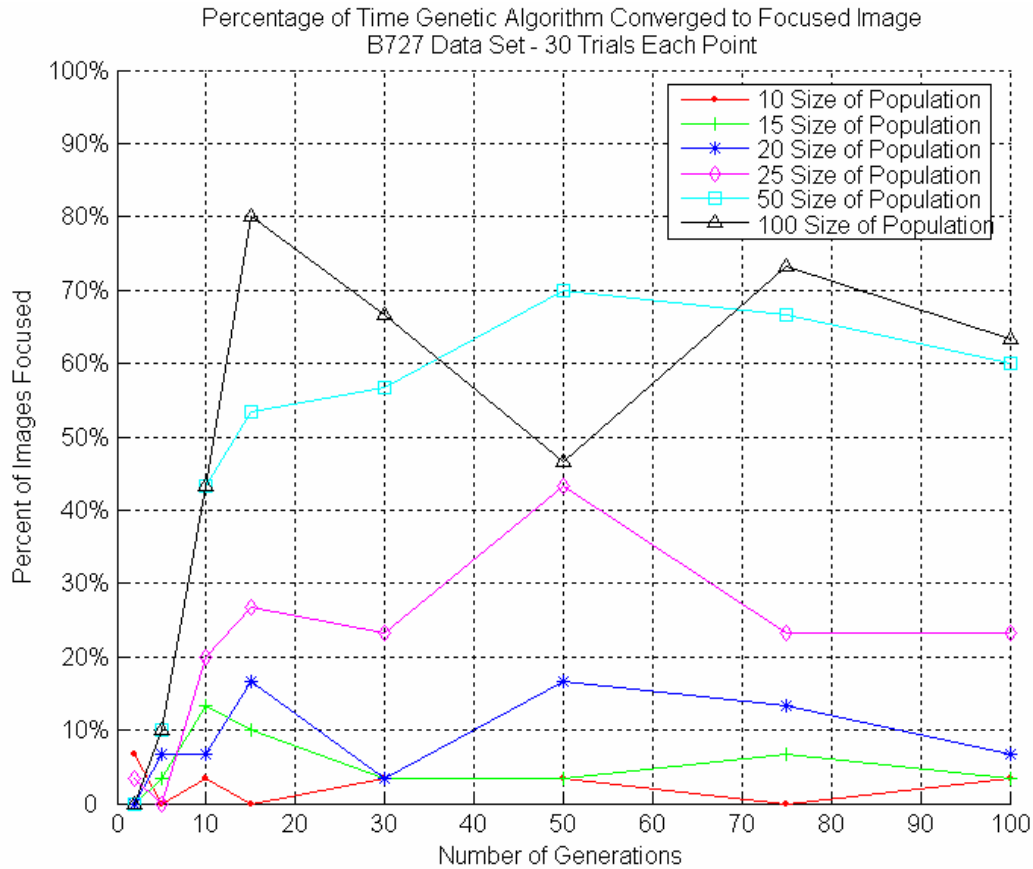


Figure 22. Convergence Graph of GA with Probability of Reinsertion of Children into New Population Equal to 80%

In order to improve the probability of convergence, a simple method which combines deterministic and random processes was chosen. When a new population is ready to be formed, 50% of the slots are filled by the best parents of the old population and the best children of the new population. The remaining slots are filled by randomly choosing from the parents and children not yet chosen for insertion into the new population. This method accomplishes two goals. First, it ensures that the best values found by the GA will continue to influence the evolution of the GA from generation to

generation which encourages local searches around these good parameter sets. Secondly, the random selection of the second half of the new population ensures sufficient genetic diversity in the population which encourages global searching of the entire solution space. This is the method that was employed in this thesis' GA. Another convergence graph was generated with this change but it, and its subsequent improvement, are discussed in the results section.

f. Determine if the Search is Complete

There are a couple of ways to determine if the GA search is complete. The first is a deterministic way where the user indicates the maximum number of generations that the search algorithm is allowed to evolve. After the last generation, the best parameter is taken and used for either translational or rotational motion compensation. The second method is to measure the number of generations that the best value found has remained unchanged. The longer this value remains stagnant, the likelihood of finding a better value becomes less and less. After a fixed number of generations that this value remains stagnant, the GA will stop and use this value for motion compensation. For this thesis, the user indicates the number of generations the GA will run since this allows for the generation of convergence graphs, which in turn is a good measure of the GA's performance and allows the GA to be compared to the performance of the PSO algorithm.

Finally, this procedure must be completed twice, once for translational and then for rotational motion compensation. If compared to the exhaustive search, the number of cost function calculations for a GA can be determined by:

$$N_{\text{cost_function_GA}} = 2 \cdot N_{\text{pop}} \cdot N_{\text{Generations}} \quad (37)$$

2. GA ISAR Results

To determine the functionality of the AJTF GA, the B727 simulated data set and several different 2048 pulse imaging intervals of the 6 – Point Delta Wing data set were used.

a. B727 Simulated Data Set

Shown in Figure 23 is the unfocused B727 and the power spectras of the two range bins used in the AJTF GA. It is easy to see from both the image and the power spectras that there is time-varying Doppler in the simulated radar data set. Figure 24

shows the image after translational motion compensation of 3rd-degree motion. Since most of the error is rotational, as stated in [3], the image is still not focused with the exception of one of the two point scatterers in range bin 31, which was changed in its spectrum from being very broad to very narrow. Figure 25 shows the final focused image. Notice that the image, after 3rd-degree rotational motion compensation, is well focused and that the scatterers are now sharp points in the frequency domain. This image was formed with a population size of 20 and was evolved for 20 generations for a total run time of 0.801 seconds for a $N_{\text{cost_function_GA}} = 800$. As given in Equation (33), an exhaustive search would have taken 2048 calculations of the cost function.

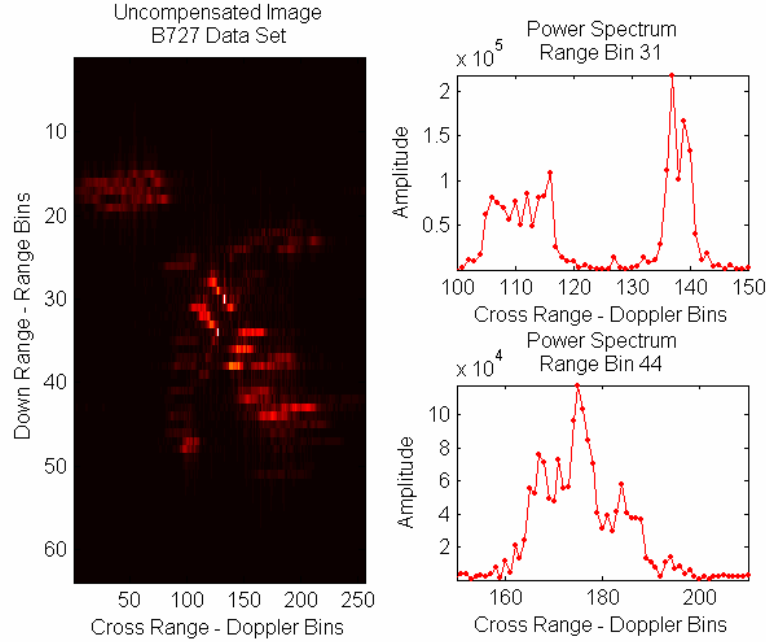


Figure 23. Unfocused B727 and the Power Spectras of Range Bins 31 and 44.

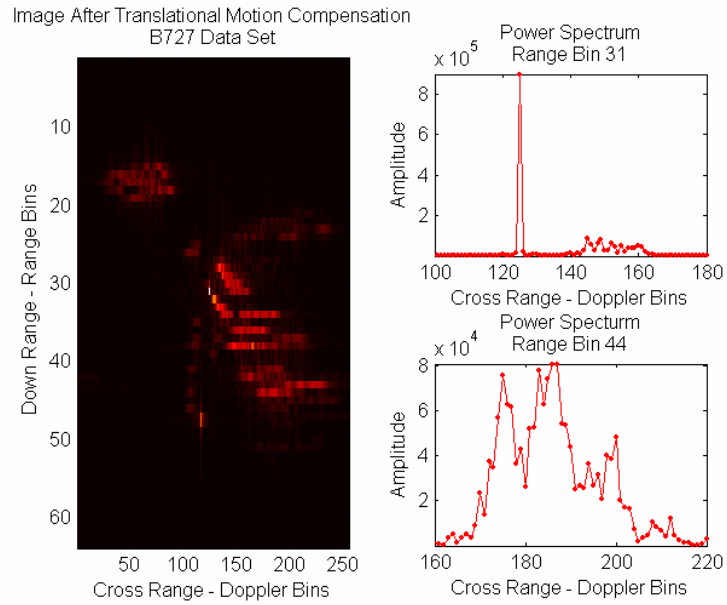


Figure 24. B727 and the Power Spectras of Range Bins 31 and 44 after Translational Motion Compensation.

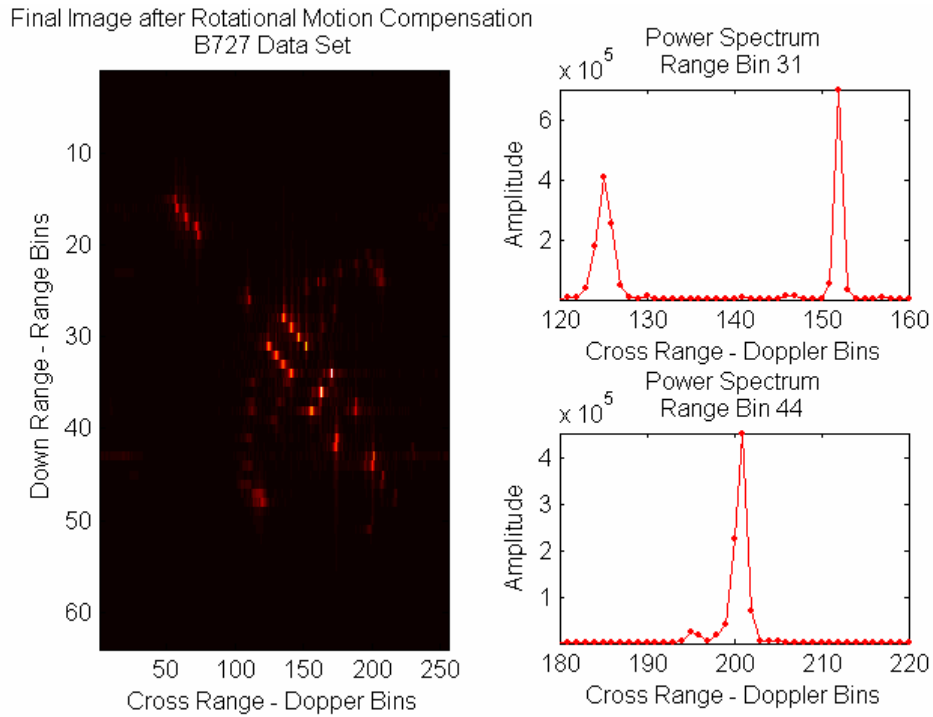


Figure 25. Focused B727 and the Power Spectras of Range Bins 31 and 44.

n order to better understand the relationship between the population size, the number of generations and the probability of convergence, a convergence graph was calculated. It was built by varying the population size, increasing the number of generations that the GA would cycle through and, after 50 tests for focus had been made for each point, measuring the percentage of times the unfocused image came into focus. Focus is measured by taking the 2D cross-correlation of the focused B727 image with the resulting image generated by the GA. Cross-correlating an unfocused image with the focused image will create a smeared 2D cross-correlation plane with several points in red as shown in Figure 26. Cross-correlating a focused image with the focused image will create a 2D cross-correlation plane with one distinct peak as in Figure 27. This can easily be detected through threshold detection.

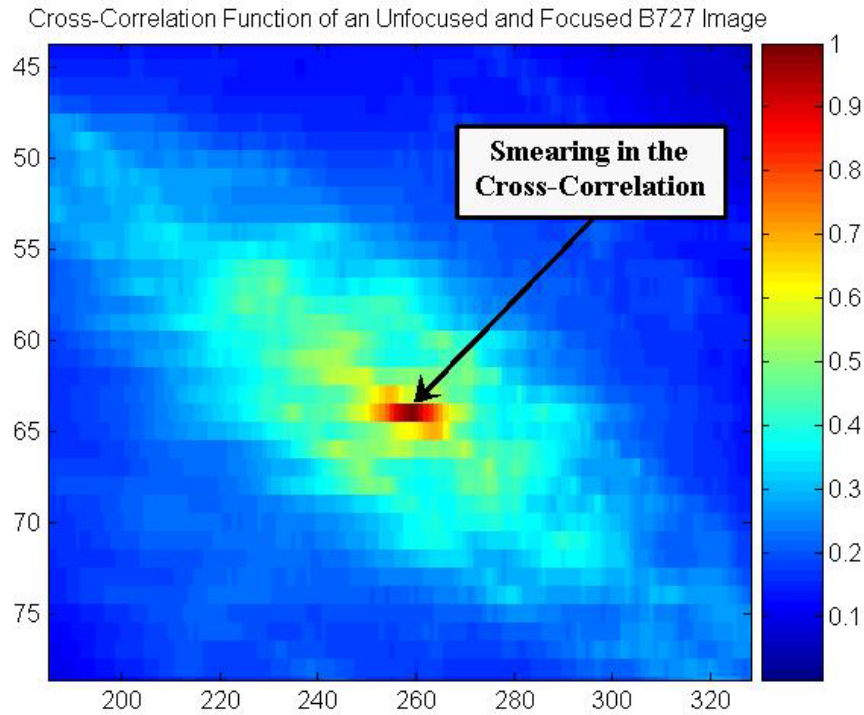


Figure 26. Cross-Correlation between a Focused and Unfocused B727 Image.

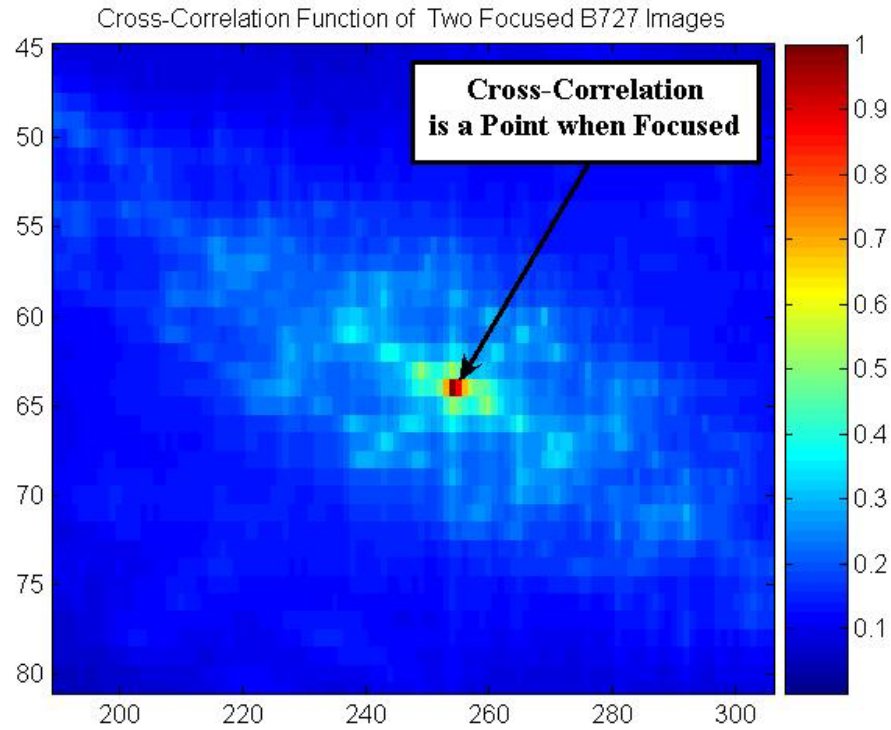


Figure 27. Correlation between Two Focused B727 Images.

Figure 28 shows the convergence graph as the percentage of images focused as a function of varying the population size and the number of generations the algorithm was allowed to evolve. Figure 29 shows the average run time of each of these trials. These graphs are very useful in determining optimal search parameters for the GA search. For example, from these graphs it can be seen that an optimal choice of parameters, for the B727 simulated data set, would be a GA search with population size of 60 which is evolved for 10 generations since this gives an 84% chance of convergence, a run time of only 0.96 seconds and only 1200 calculations of the cost function. The requirement to calculate the cost function 1200 times is only 58.5% of the minimum cost function calculations needed by the exhaustive search.

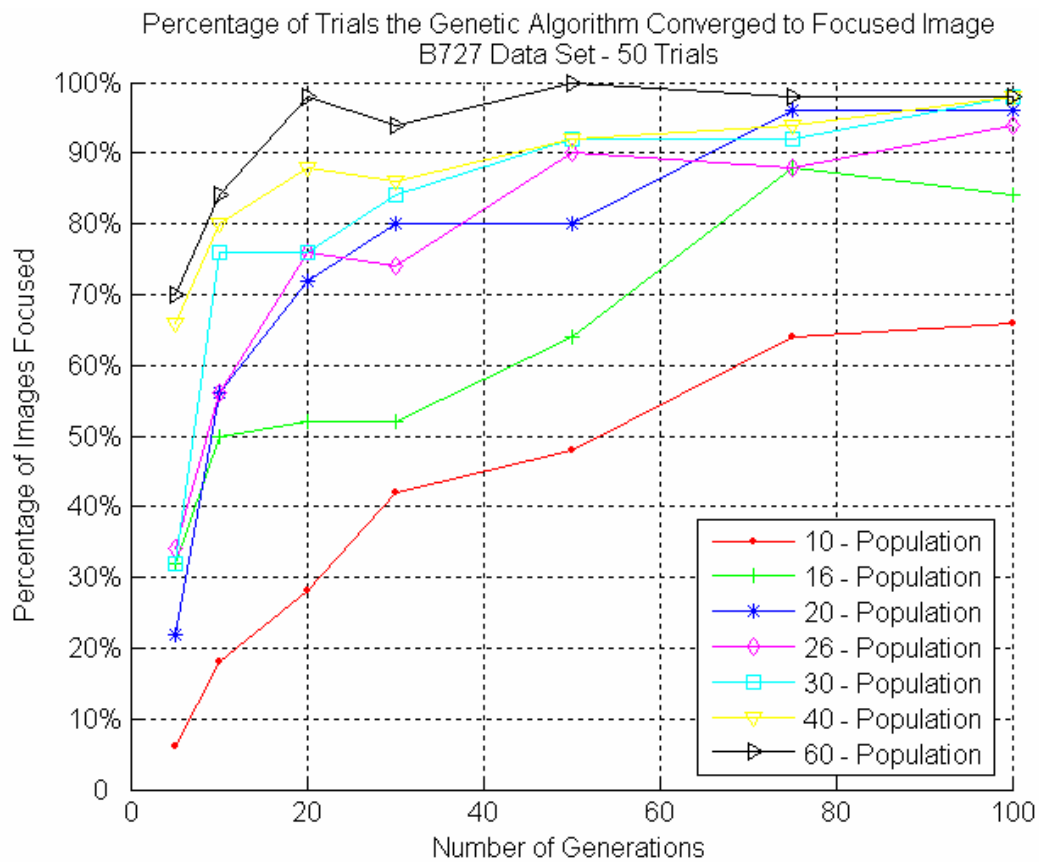


Figure 28. Percentage of Images Focused Using the Genetic Algorithm as a Function of Population Size and Number of Generations – B727 Data Set.

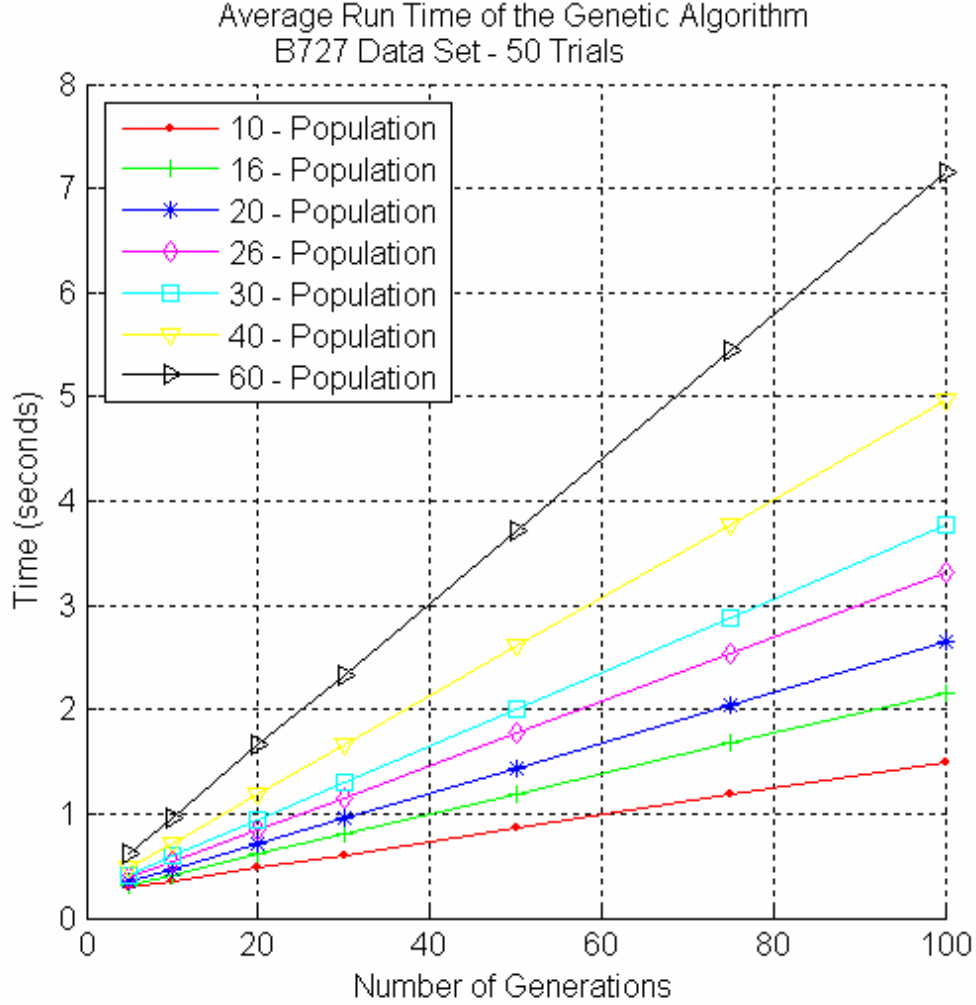


Figure 29. Average Run Time of the Genetic Algorithm as a Function of Population Size and Number of Generations – B727 Data Set.

b. 6 – Point Delta Wing Experimental Data Set

The 6 – Point Delta Wing experimental data set, with 2048 (2^{11}) pulses in the cross-range, represents a more challenging focusing problem as it does not always exhibit ideal motion error characteristics like the B727 simulated data set. Though the assumptions of rigid body, two-dimensional motion still, for the most part, hold for this data set, there are many imaging intervals for which they do not hold. For the moment, the discussion of results will be limited to four imaging intervals and only 2nd-order motion error (i.e., a 1-dimensional solution space search for parameter f_{12} and f_{22}) where the GA is capable of finding a parameter that will focus the image. For comparison, the

number of cost function calculations for translational and rotational motion compensation using the exhaustive search, as in Equation (32), is 8196 calculations of the cost function.

The first imaging interval is imaging interval 15 and is shown in Figure 30. This imaging interval has only one focused point scatterer which can be placed in its proper cross-range bin by the fast Fourier transform. The remaining scatterers possess time-varying Doppler and are badly blurred in the cross-range. The GA AJTF algorithm with a population of size 30 can find the search parameter to focus the image after 50 generations which equates to 3000 calculations of the cost function or 37% of the calculations when compared to the exhaustive search. Notice that the power spectrum in the unfocused image is spread in Doppler while the focused image has a power spectrum with sharp points.

The second imaging interval is interval 19 and is shown in Figure 31. Again, five of the scatterers are out of focus in the cross-range. The AJTF GA is successful in finding the first order parameters, f_{12} and f_{22} , and focusing the image. All six point scatterers can be clearly identified in the image. Figure 32 shows imaging interval 23 whose unfocused image has significant motion error as only one corner reflector is in its proper Doppler bin. The focusing algorithm removes the error and focuses the image in the cross-range. The sixth reflector is located in range bin 20. However, as it is shadowed by a reflector on the leading edge of the 6 – Point Delta Wing, in range bin 26, it is below threshold and not visible. Again both the imaging intervals were formed with 3000 calculation of the cost function.

The final imaging interval to be presented is imaging interval 14 in Figure 33. The original image is very badly blurred in the cross-range. The AJTF GA is able to focus most of the unfocused scatterers extremely well. There still is some blurring, particularly in range bin 25 but also a small amount in range bins 26, 21 and 28. The inability of the AJTF algorithm to remove all blurring is most likely due to a violation of one of the basic assumptions used to formulate the mathematical model of the AJTF, most likely the presence of a very small degree of 3D motion.

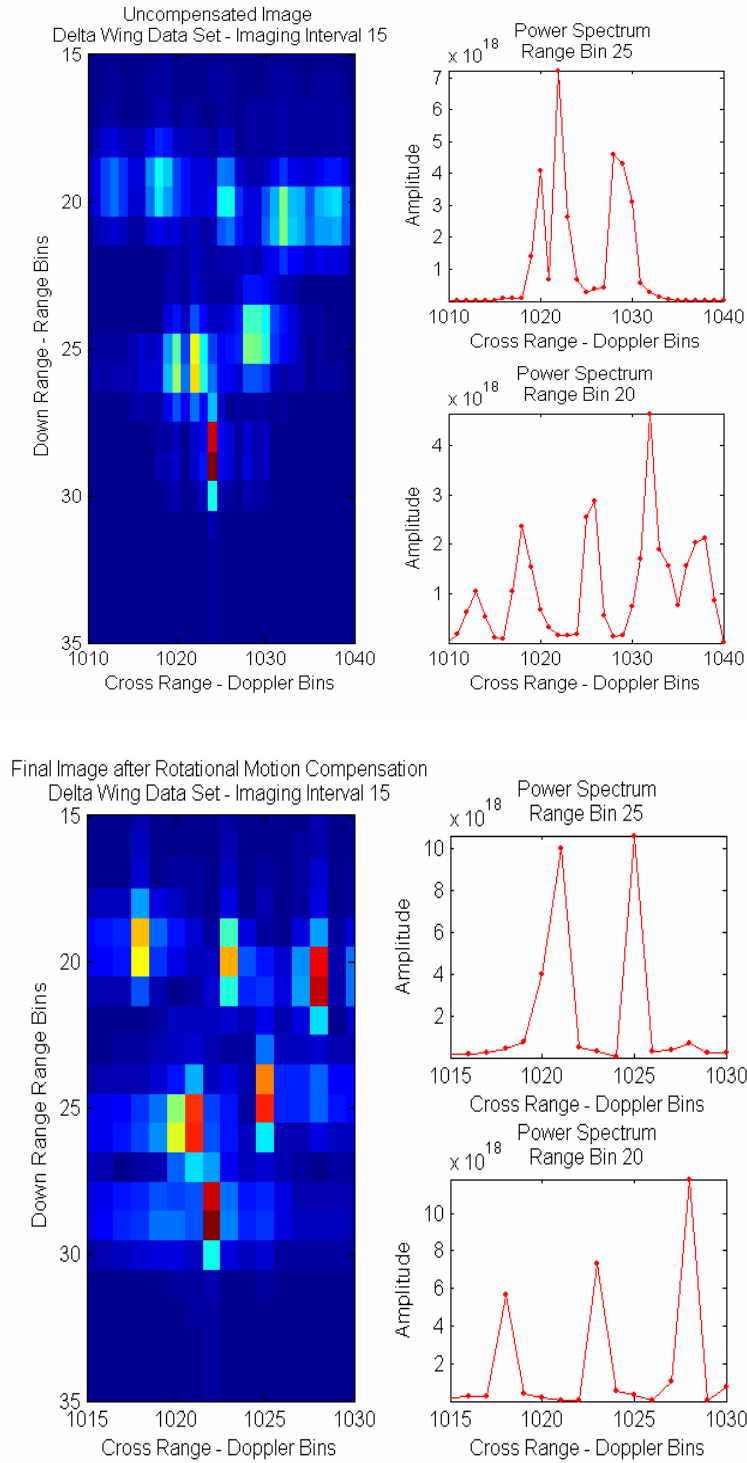


Figure 30. Uncompensated (top) and Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 15 with 2048 Pulses in the Cross-Range – GA.

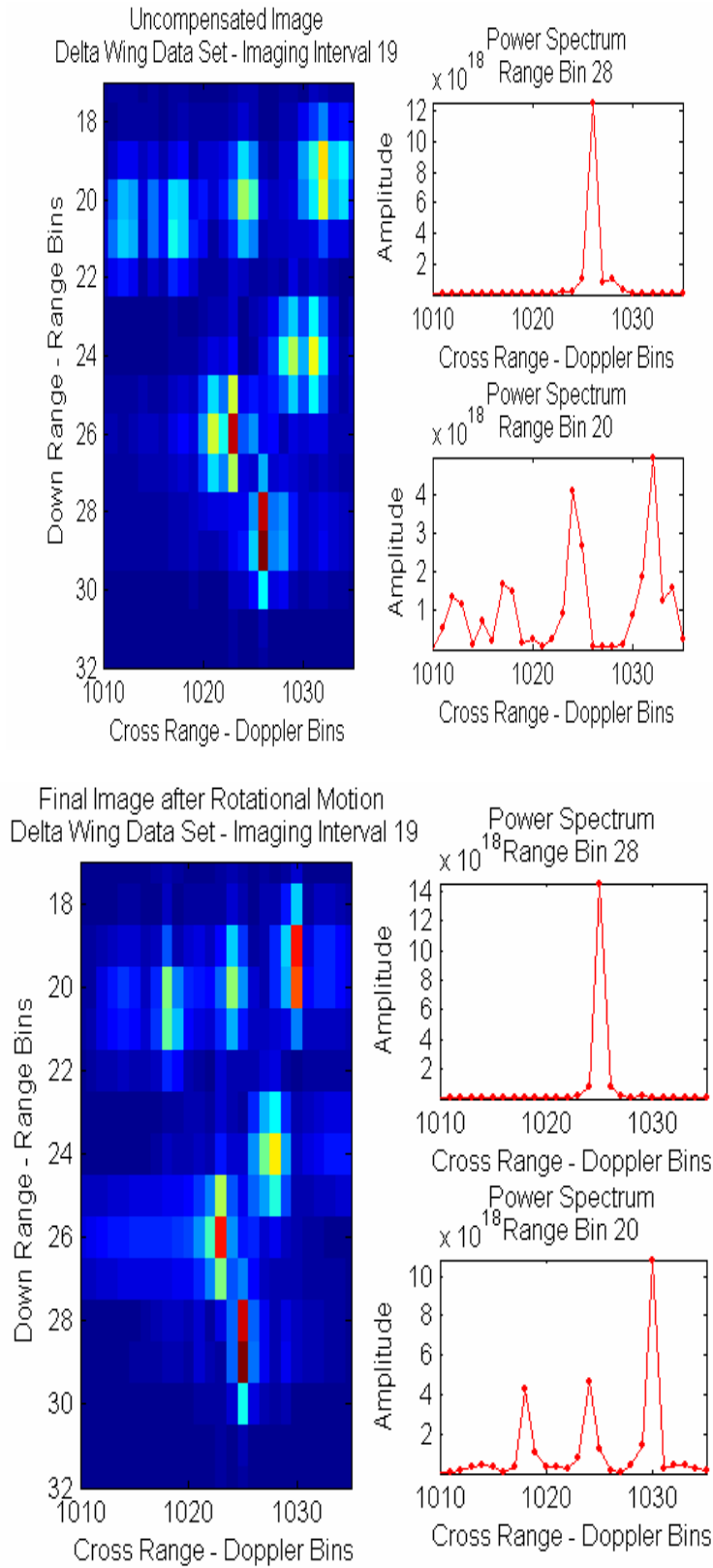


Figure 31. Uncompensated (top) and Motion Compensated 6 – Point Delta Wing ISAR Image – Imaging Interval 19 with 2048 Pulse in the Cross-Range – GA.

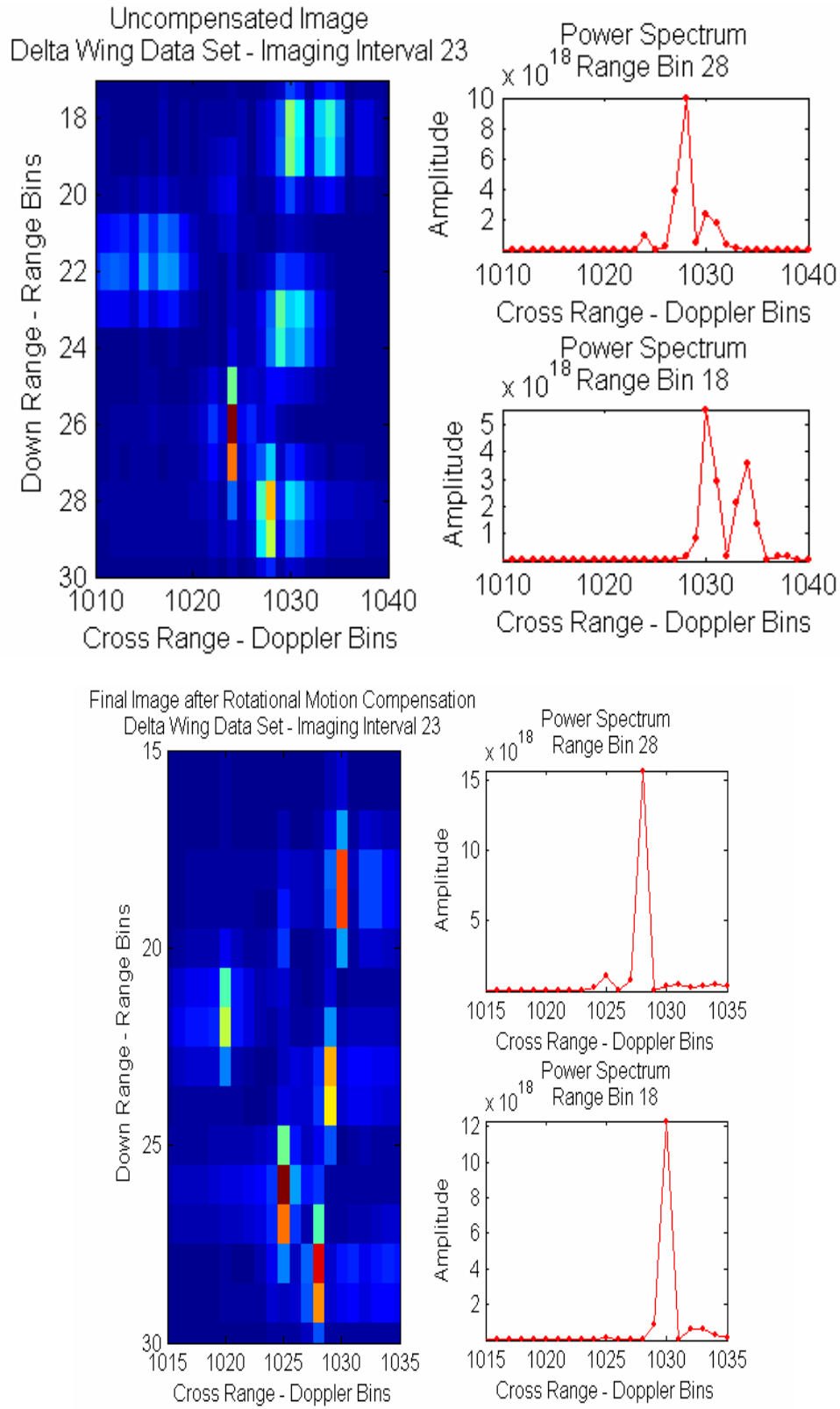


Figure 32. Uncompensated (top) and Motion Compensated Delta Wing ISAR Image – Imaging Interval 23 with 2048 Pulses in the Cross-Range – GA.

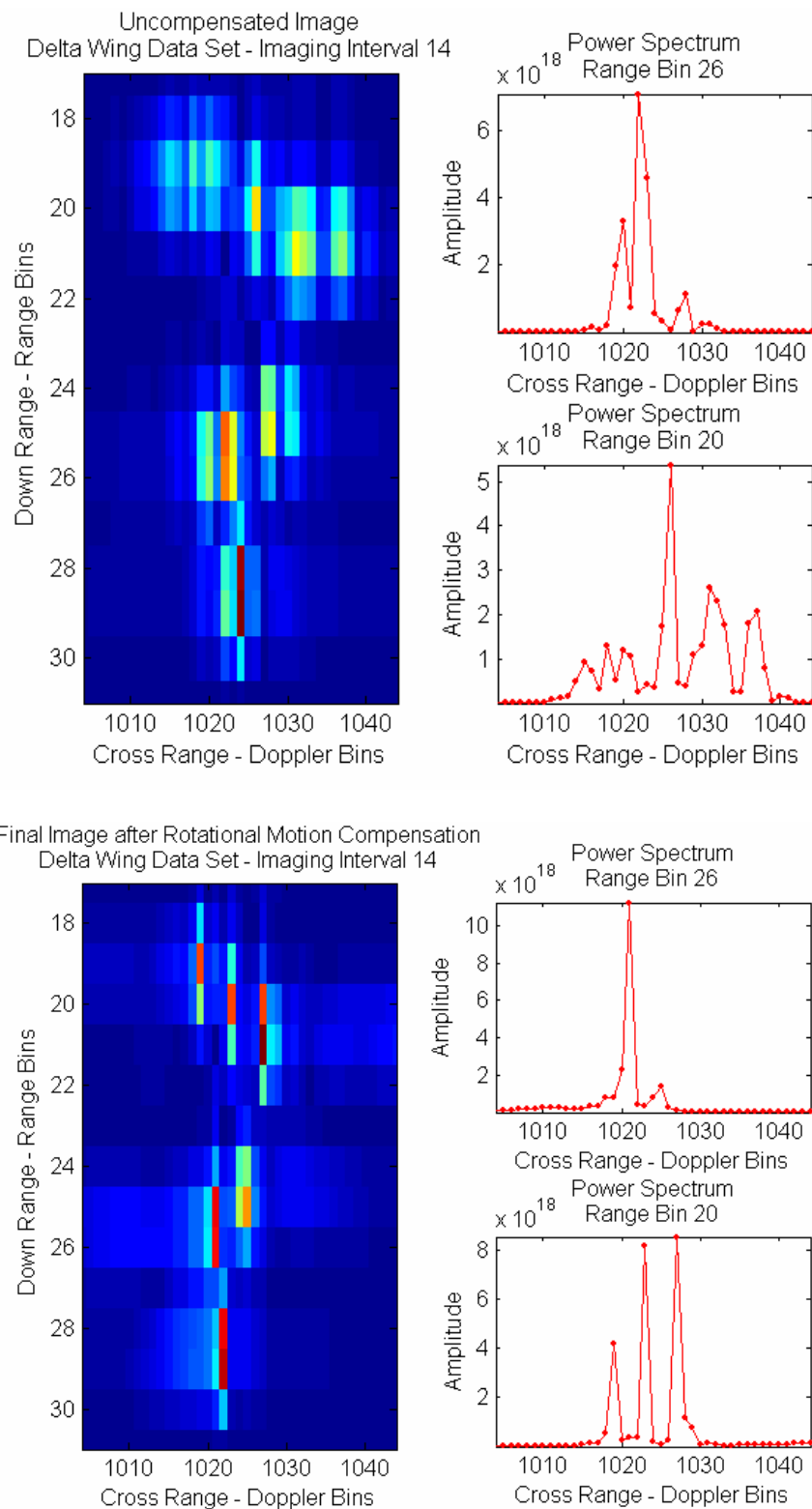


Figure 33. Uncompensated (top) and Motion Compensated Delta Wing ISAR Image – Imaging Interval 14 with 2048 Pulses in the Cross-Range – GA.

As with the B727 data set, a convergence graph for the Delta Wing data set was calculated in order to determine the best combination of population size, number of generations in the run time and the resulting probability of convergence. Imaging interval 15 was used as the test data set to be focused by the GA. The resulting image was then compared to the focused version of this imaging interval via the 2D cross-correlation function following the methodology as explained for the B727 data set.

The convergence graph is shown in Figure 34. From the graph, it can be seen that for an 80% probability of convergence, a population size of 50 and a run of 50 generations is necessary. This translates into 5000 calculations of the cost function, or 61% of what is necessary for the exhaustive search. It is shown in Figure 35 that the average run time for a GA with this population size and 50 generational evolutions is 22 seconds.

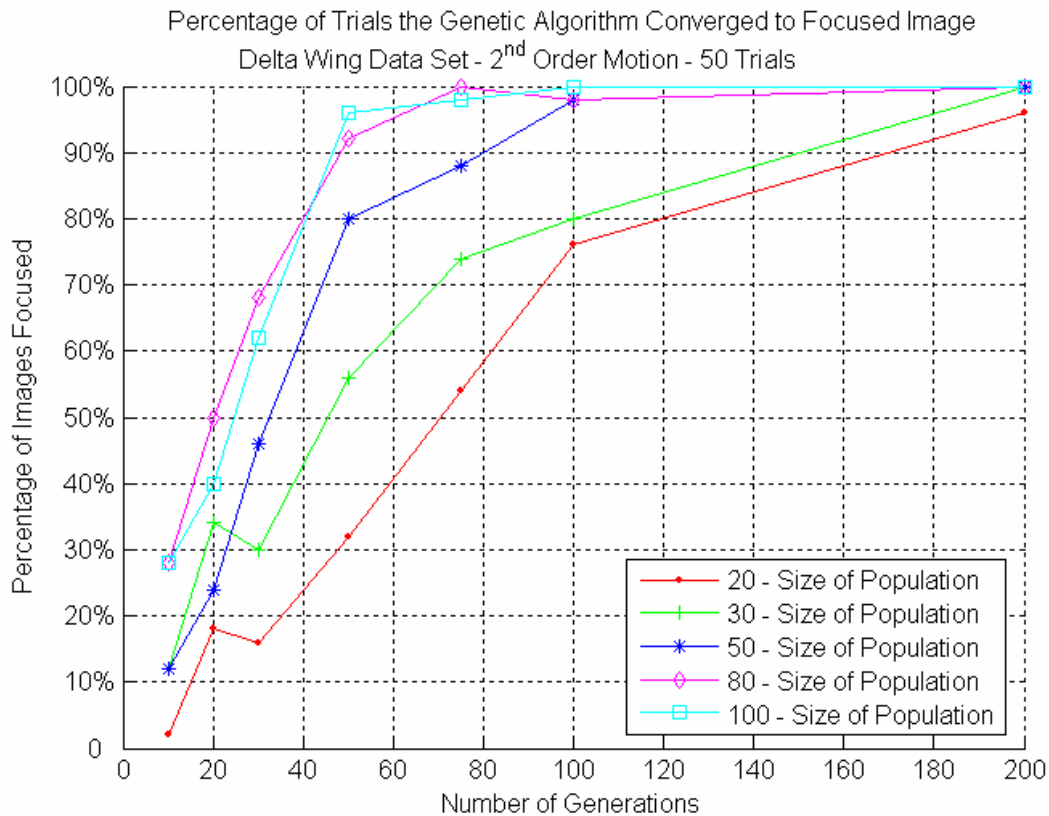


Figure 34. Percentage of Images Focused Using the Genetic Algorithm as a Function of Population Size and Number of Generations: 6 – Point Delta Wing Data Set.

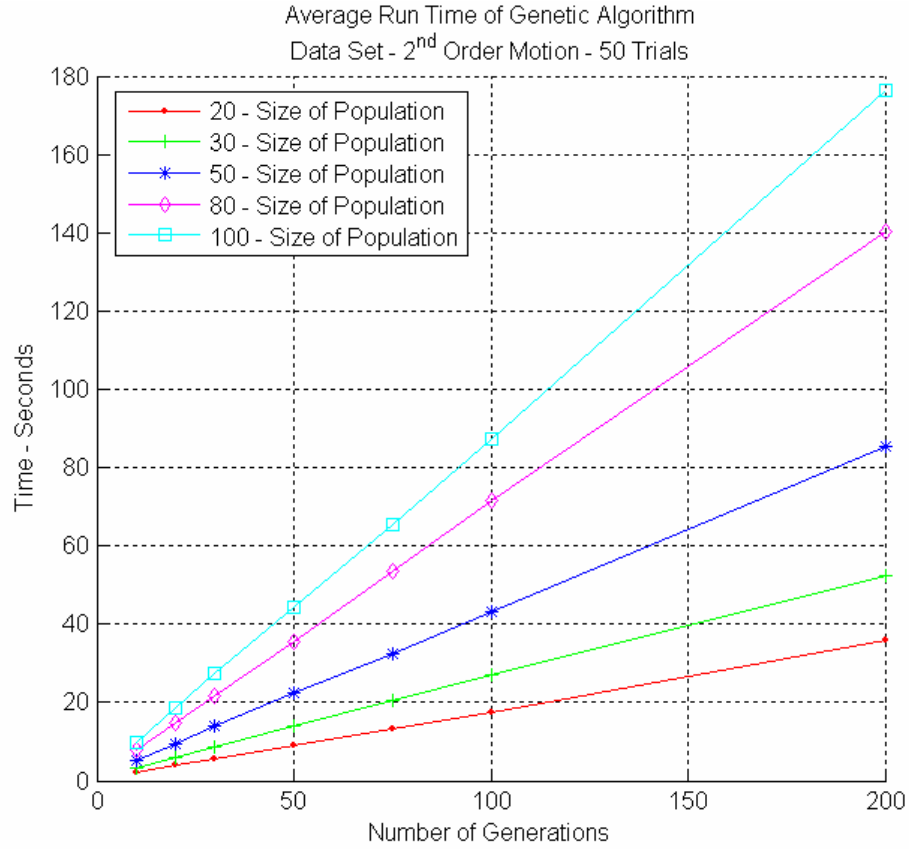


Figure 35. Average Run Time of the Genetic Algorithm as a Function of Population Size and Number of Generations: 6 – Point Delta Wing Data Set.

C. PARTICLE SWARM OPTIMIZATION PARAMETER SEARCH

The particle swarm optimization (PSO) algorithm was introduced in [7] and finds its roots in the fact that the way that fish school and bees swarm is optimal in nature and can be adapted for parameter searches to optimally solve engineering problems. In the case of the ISAR focusing problem, it is ideally suited to finding the optimal values for the search parameters, $\{f_{12}, f_{13}, \dots\}$ and $\{f_{22}, f_{23}, \dots\}$, as it displays the same search characteristics as GA, in particular, the ability to rapidly converge towards a global maximum. Figure 17 and the arguments that accompany it about initial rapid improvement to the found global maximum followed by smaller improvements after many cycles applies equally to the PSO as explained for the GA. The driving force behind the particle swarm optimization is that each particle in the swarm knows the fitness of its current location, the best location it has ever encountered, the Local Best, and the best location ever en-

countered by any particle of the swarm, the Global Best. It is this knowledge that forms the search pattern of the solution space in order to find the optimal solution.

1. Sequence of the Particle Swarm Optimization Algorithm

Like the GA, the PSO follows a set sequence that repeats until done. Figure 36 graphically describes each of the steps in the PSO as it applies to ISAR focusing. The following paragraphs describe each step in detail.

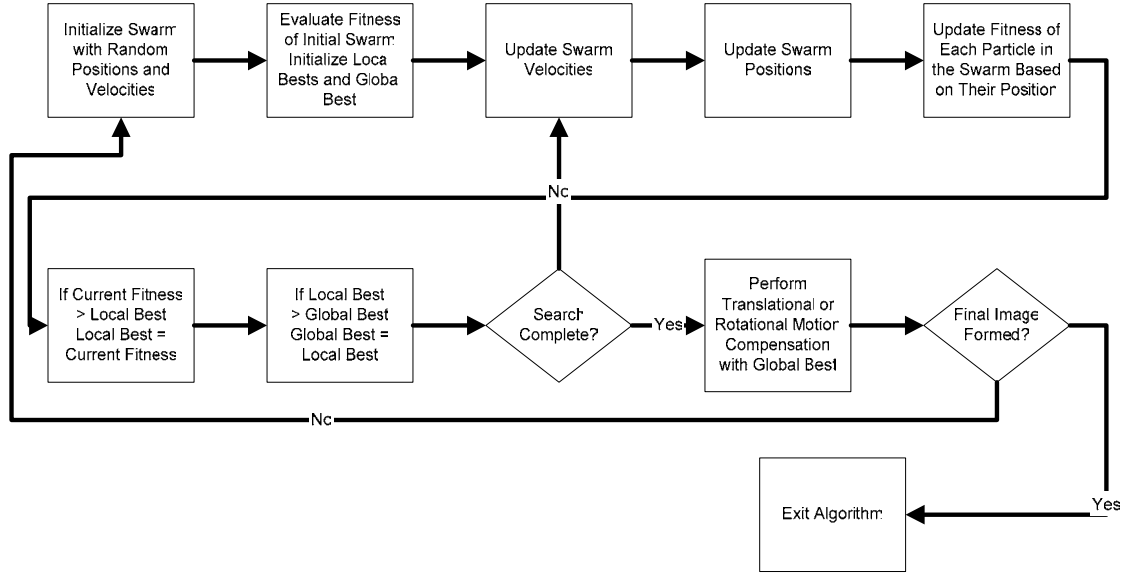


Figure 36. The PSO Flowchart (After [7].)

a. Defining the Particles of the Swarm

The swarm in a PSO is identical in definition to the population in a GA, as shown in Table 1. Each particle in the swarm is analogous to a parent and holds the search parameters $\{f_1, f_2, f_3 \dots\}$, as determined by the degree of motion error. Each particle is initialized as a uniform random variable from $+N$ to $-N$, where N is the number of pulses in the cross-range. At the same instance of defining the swarm, each particle is designated an initial velocity that is uniformly random between $-2N$ and $+2N$. The maximum velocity, $|v_{\max}|$, that any particle can possess is $2N$. Figure 37 shows a swarm of a 3rd-degree parameter search after initialization. As expected, it is characterized by the uniform spread of the particles through the expanse of the solution space.

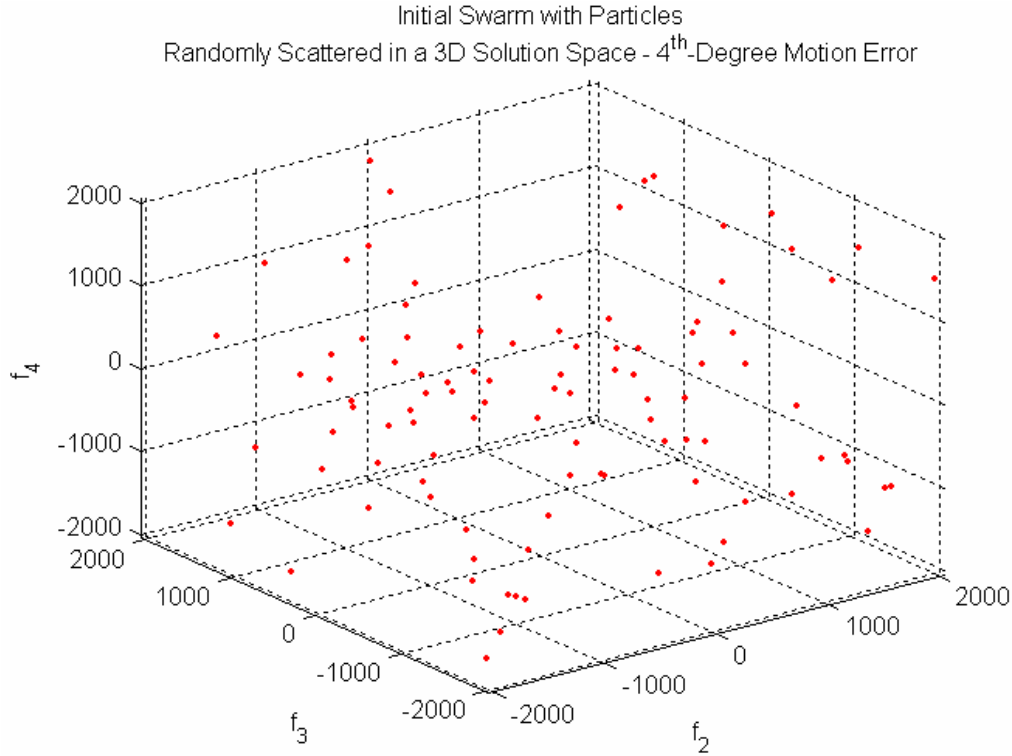


Figure 37. Example of an Initial Swarm in a Solution Space for 4th-Degree Motion Error.

b. Assessing Initial Fitness of the Swarm and Defining the Local Best Vector and Global Best Particle

Now that the initial swarm has been defined, each particle is assessed for its fitness as defined by the cost function in Equation (28) in a process that is identical to that described in the GA section of this chapter. In fact, the MatLab code that defines the fitness is identical for the GA and the PSO. Once the fitness is known, the best particle in the swarm becomes the Global Best which holds the location of this particle and its fitness score. The Local Best vector is initialized with the current particle positions and their current fitness since this position is each particle's first position and, therefore, by default each particle's best position they have ever encountered.

c. Updating the Particle Velocities and Positions

Now that each of particles has a fitness score assigned to it and its velocity initialized, the particle velocities can be updated based on the current location of each individual particle, its Local Best particle and the Global Best particle of the swarm. To do this Equation (3) in [7] is used:

$$v_n = K \left(v_n + \varphi_1 \cdot rand \cdot (f_{Local,n} - f_n) + \varphi_2 \cdot rand \cdot (f_{Global,n} - f_n) \right) \quad (38)$$

where v_n is the particle's current velocity, f_n is the particle's current location in the n^{th} dimension of the solution space and $f_{Local,n}$ and $f_{Global,n}$ are the Local and Global best positions, respectively, in the n^{th} dimension. The PSO parameters φ_1 , φ_2 and K are parameters that, as stated by [7], were determined by extensive research to be optimal when set to 2.8, 1.3 and 0.729, respectively. The values of φ_1 and φ_2 controls the balance between the attraction of the particle to the local and global bests. With $\varphi_1 > \varphi_2$, the particle is more prone to be attracted towards searching the area surrounding the particle's local best. The constriction factor, K , can be compared to inertia and it effectively assists in the deceleration of the particle as convergence occurs. *Rand* is a uniformly distributed random variable between 0 and 1 that serves the purpose of adding randomness to this attraction and leads to a more complete spanning of the solution space. We can see from Equation (38) that, as the separation between the particle's current position and the swarm's Global Best and its Local Best expands, the greater effect they will have on the particle's velocity and subsequently its search direction. Also, the search pattern of the entire swarm is instantaneously affected every time the Global Best changes. Reference [7] states that this equation is analogous to a bee's desire to keep searching for food in the direction that it is going but being instinctively pulled towards the best location it has ever found (Local Best) and the best location ever found by any member of the swarm (Global Best).

To update the particle's positions, it is simply a matter of applying Equation (39) to each of the search parameters [7]:

$$f_n = f_n + v_n t. \quad (39)$$

Typically t is taken to be equal to 1 so, to generate the next cycle of the swarm, it simply becomes a matter of adding the velocities, v_n , to each appropriate search dimension's current positions. The effect that the velocity update has on the particles is shown in Figure 38.

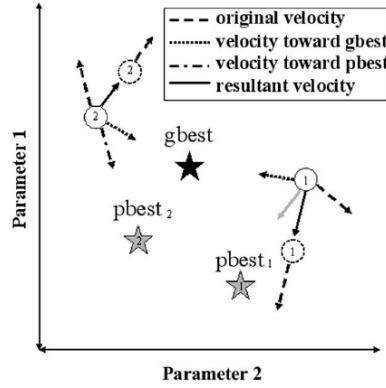


Figure 38. Graphical Illustration of the Velocity and Position Update Process. Note that $gbest$ = Global Best and $pbest$ = Local Best (From [7].)

When discussing velocity and position updates, it is necessary to discuss the boundary conditions of the solution space since particles left to themselves will depart the solution space where meaningful solutions reside. Since we know from previous discussion that our search parameters, $\{f_2, f_3, \dots\}$, will possess values between $\pm N$, each dimension of the solution space has a boundary at $\pm N$ (where N is again the number of pulses in the cross-range). Reference [7] states that there are three ways to deal with particles that exceed the boundary of the solution space:

- i. **Absorbing Walls:** When a particle exceeds a boundary in a dimension of the solution space, the velocity in this dimension is immediately zeroed and its position is set to the boundary. The particle will then rejoin the swarm when its velocity is updated during the next cycle. Extreme care must be taken when using this boundary condition since a serious problem that is not mentioned in [7] may occur. If the situation exists where a local maximum exists at the boundary and if this position is simultaneously the Local Best and the Global Best position, the velocity will remain stuck at zero and the particle will stay fixed to the boundary. A condition was observed where all the particles in a swarm became stuck to the boundary and the swarm did not converge to the true global maximum and the focused image after motion compensation was very poor.
- ii. **Invisible Walls:** Here the idea is to let the swarm cycle without interruption and any particle that strays beyond the boundaries of the solution space is not evaluated against the cost function. Instead it is immediately assigned a fitness equal to

zero. As such, this particle outside the boundaries will not encounter any new local or global maximums. This will cause the velocity update equation, as in Equation (38), to lead the particle back into the solution space without interruption to the particle's natural search pattern.

iii. Rebounding Walls: When a particle passes a boundary, its position is immediately set to the boundary and it is given a velocity that points away from the boundary. This new velocity is equal to

$$v_n = \pm 0.1 \cdot rand \cdot |v_{\max_n}| \quad (40)$$

where v_{\max_n} is the maximum velocity of a particle in the n^{th} direction, *rand* is the usual uniformly distributed random variable between 0 and 1 and the direction is set such that it is heading away from the boundary. The 10% scaling factor keeps the particle near the boundary, since that is where it was headed, while still allowing the particle to retain momentum and avoid the problem of sticking to the boundary, as explained above in the rebounding walls boundary condition.

Reference [7] goes into greater detail on the strengths and weaknesses of each boundary condition. However, the PSO algorithm written for this thesis uses the rebounding walls option since it keeps all the particles within the solution space. This creates the condition where each particle will have a new position to evaluate against the cost function for each cycle, a critical factor when speed of convergence is important. Also, there is no danger of a problem with the boundary, as explained for the absorbing wall boundary condition.

Finally, when updating the velocity of the particles, it is necessary to ensure that no particle exceeds $|v_{\max}|$ in either the positive or negative direction. Reference [7] states that v_{\max} has been determined to be optimal when set to equal the dynamic range of the solution space in a dimension. This is why, as previously stated, $|v_{\max}| = 2 \cdot N$. When a particle is observed exceeding the maximum velocity, it is simply set equal to $\pm v_{\max}$, with the direction of velocity preserved.

d. Updating Particle Fitness and Reassigning Local Best Vector and Global Best Particle

Now that the new positions of each of the particles have been defined, their current fitness values are assessed as previous discussed. If the fitness score of any of the particles exceeds that of the Global Best, it becomes the new Global Best. In addition, each of the particles current fitness is assessed against their individual Local Best. If the fitness of their current position exceeds their Local Best, the current position becomes the Local Best. In this way, as new Local Bests and new Global Bests are found, the search pattern of each particle and the entire swarm is updated to bias their search towards these areas.

e. Termination of the Search

At the termination of the search, the swarm should resemble the swarm shown in Figure 39 and should have converged to the solution space's global maximum. The PSO is terminated after a fixed number of cycles as chosen by the user, which is the same way that this thesis' GA is terminated. As with the GA, terminating the search in this manner adds a deterministic quantity to a random search which allows the PSO to be evaluated for performance and compared to the GA in terms of probability of convergence. Similar to this thesis' GA, on completion of the last cycle, the Global Best particle is taken and used for translational or rotational motion compensation as applicable. Also, the number of cost function calculations of the PSO algorithm can be determined using Equation (37), where the variable names have been altered to conform to the conventions of a PSO:

$$N_{\text{cost_function_Swarm}} = 2 * N_{\text{particles}} * N_{\text{cycles}}. \quad (41)$$

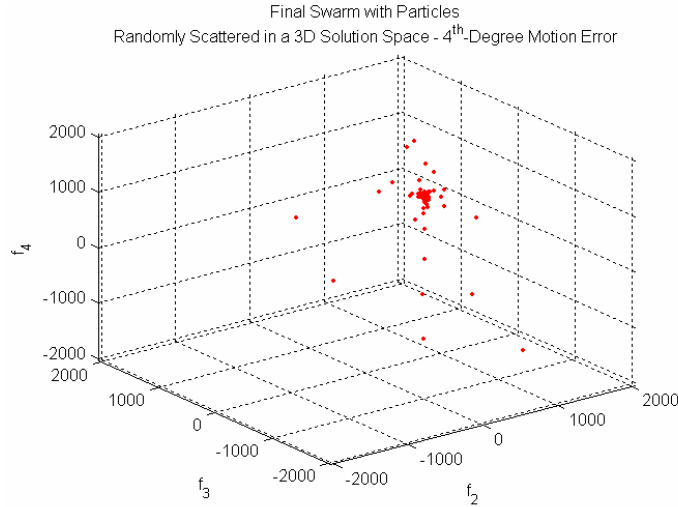


Figure 39. Example of a Final Swarm in a Solution Space for 4th-Degree Motion Error.

2. PSO ISAR Results

To determine the functionality of the AJTF PSO algorithm, the B727 simulated data set and the four 2048 pulse imaging intervals of the 6 – Point Delta Wing experimental data set, were used. These four imaging intervals were the same intervals that were used in the GA ISAR results.

a. B727 Simulated Data Set

A focused B727 image is found in Figure 40 with the spectra of the range bins used for focusing shown on the right of the image. It was focused by the PSO algorithm in 0.681 seconds after 20 cycles and with a swarm size of 20, or 800 calculations of the cost function. As with the GA, it is much more important how this algorithm behaves in a consecutive series of runs rather than just in one run instance. Using the PSO algorithm, a convergence and run time graph was constructed to verify algorithm performance when tasked to focus this simulated data set. The convergence graph is shown in Figure 41 while the average run time graph is shown in Figure 42. These two graphs show that an optimal selection of parameters could be 10 particles and 50 cycles as this produces an 88% chance of convergence, a run time that is below 1 second and requires 49% of the required cost function calculations of an exhaustive search. If probability of convergence was of critical importance, the parameters of 26 particles and 30 cycles could be chosen. These parameters yield a 94% chance of convergence at a cost of a run

time of 1.2 seconds and 76.1% of the cost function calculations as compared to the exhaustive search.

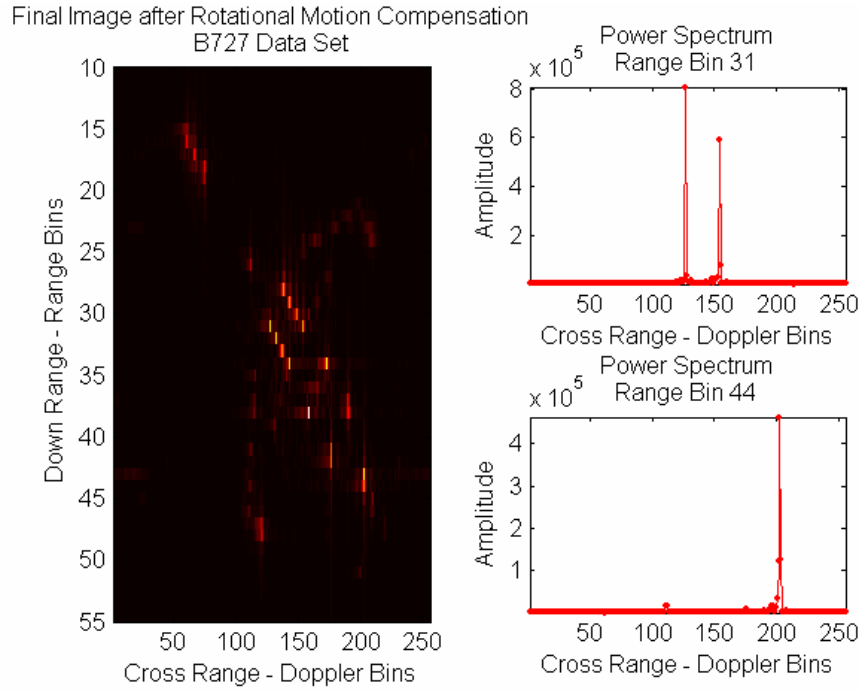


Figure 40. B727 Data Set Focused by the PSO Algorithm.

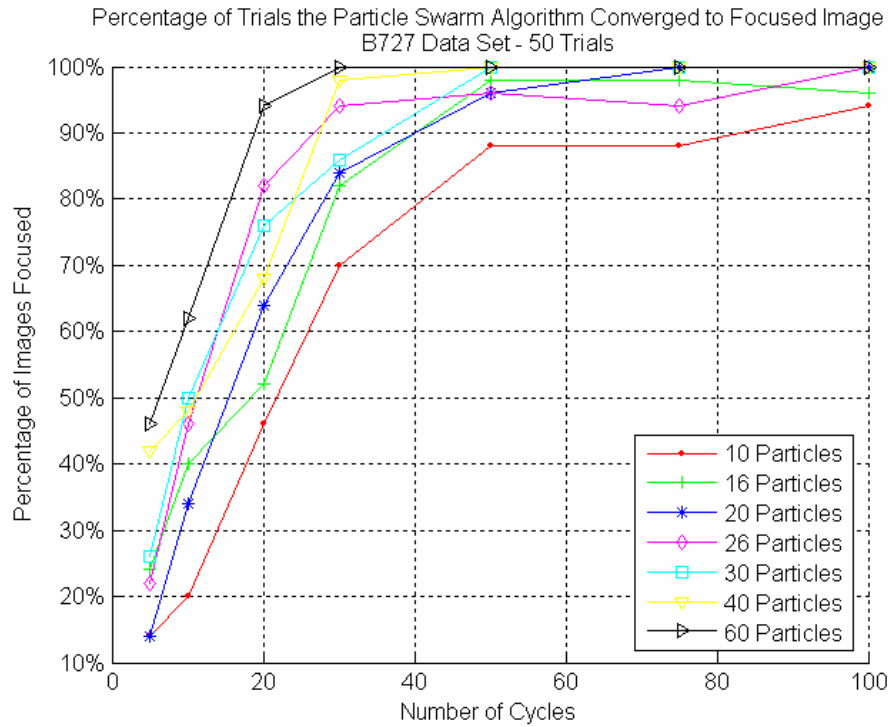


Figure 41. Percentage of Images Focused Using the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles – B727 Data Set.

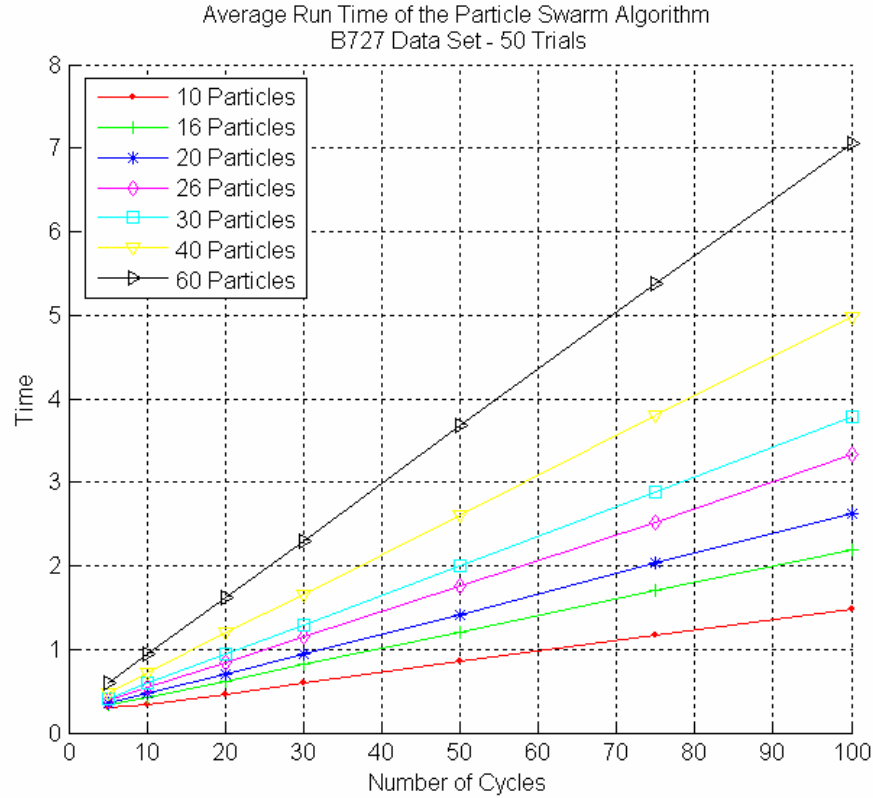


Figure 42. Average Run Time of the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles.

b. 6 – Point Delta Wing Experimental Data Set

The Particle Swarm Algorithm was tested against four 2048-pulse imaging intervals of the 6 – Point Delta Wing experimental data set in order to ensure that it could focus real radar data. These four imaging intervals, 15, 19, 23 and 14, are the same intervals that were used in the GA section of this chapter and are shown in Figures 43 – 46. The focused images, which were generated by the PSO algorithm, are exactly the same as those generated by the GA. The images were typically formed with 1200 calculations of the cost function over a run time of 6 seconds. Again, to better understand the relationship between swarm size, number of particles and their effect on the probability of convergence and run time, a convergence graph and a run time graph were constructed and are displayed in Figures 47 and 48.

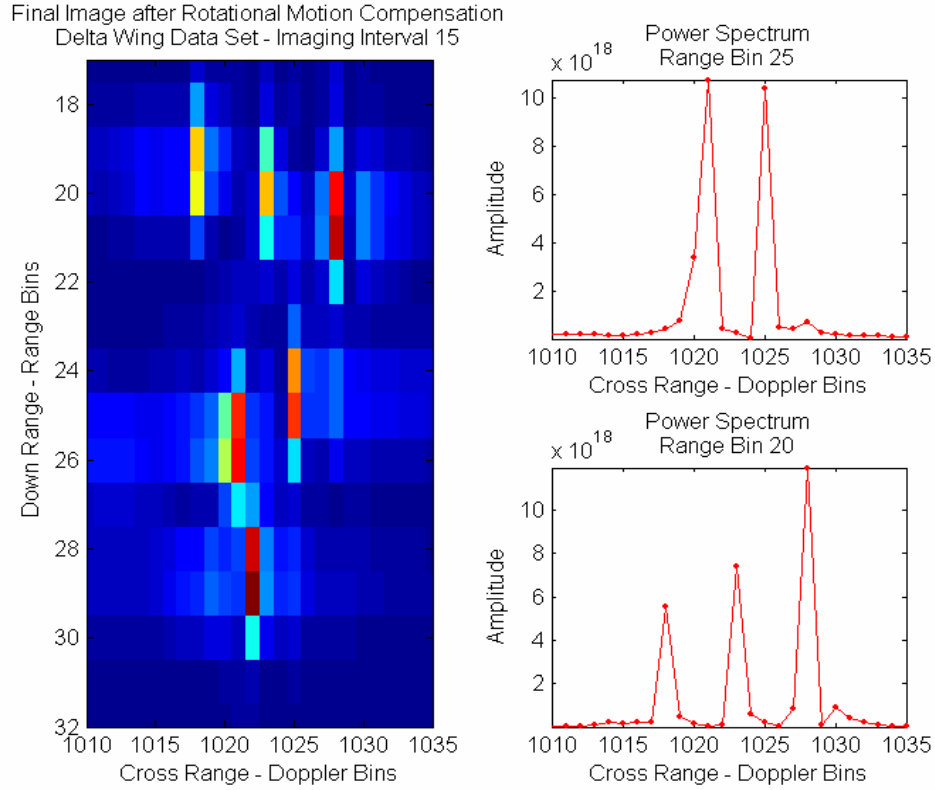


Figure 43. Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 15 with 2048 Pulses in the Cross-Range – PSO.

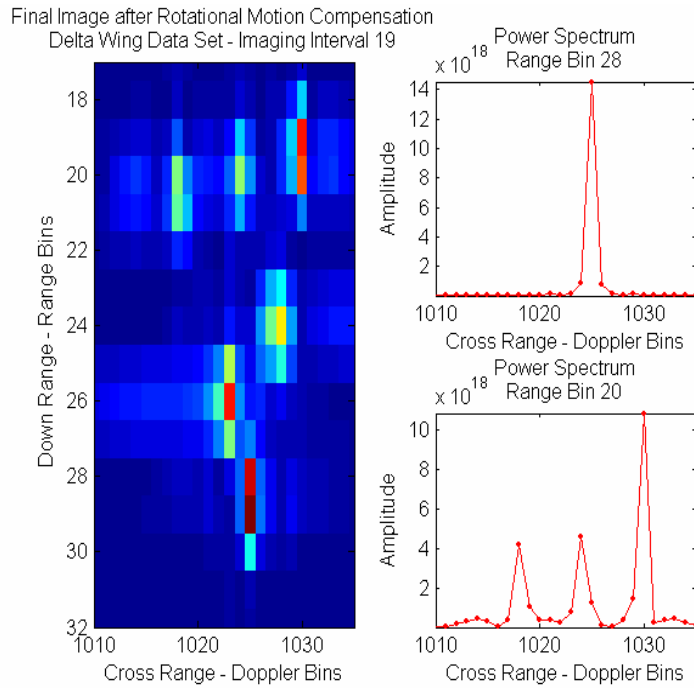


Figure 44. Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 19 with 2048 Pulses in the Cross-Range – PSO.

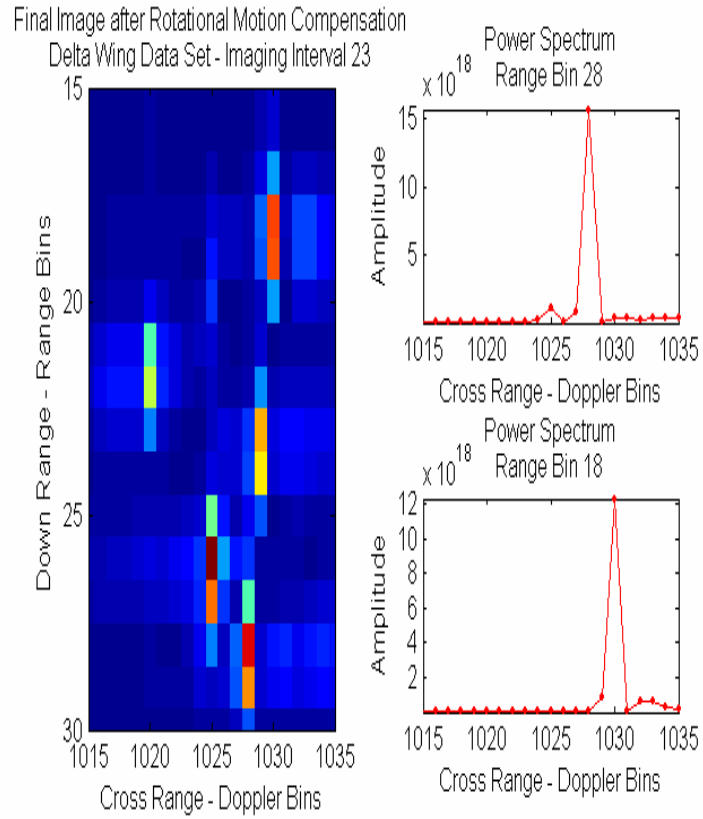


Figure 45. Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 23 with 2048 Pulses in the Cross-Range – PSO.

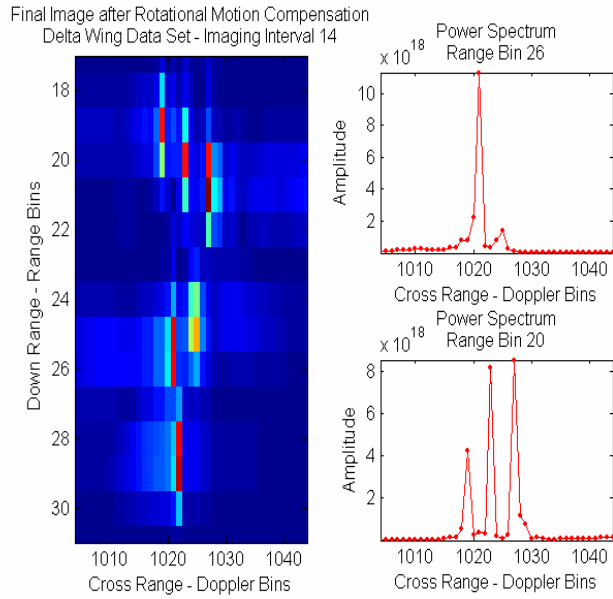


Figure 46. Motion Compensated 6 – Point Delta Wing ISAR Image – Image Interval 14 with 2048 Pulses in the Cross-Range – PSO.

From Figures 47 and 48, it can be seen that the PSO algorithm performs very well against the 6 – Point Delta Wing experimental data set and for an 84% probability of convergence, it requires an average run time of only 14.3 seconds and 3000 calculations of the cost function. This represents 37% of the cost function calculations required by the exhaustive search. Also, Figures 47 and 48 shows that the PSO is capable of performing extremely well, even with a small number of particles in the swarm. For example, using 20 particles and running the algorithm for 50 cycles leads to a total of 2000 calculations of the cost function (24% of the exhaustive search), a 9-second run time and a 98% chance of convergence to focus. This shows that this algorithm possesses both speed and reliability even against a real radar image with a large number of pulses in the cross-range.

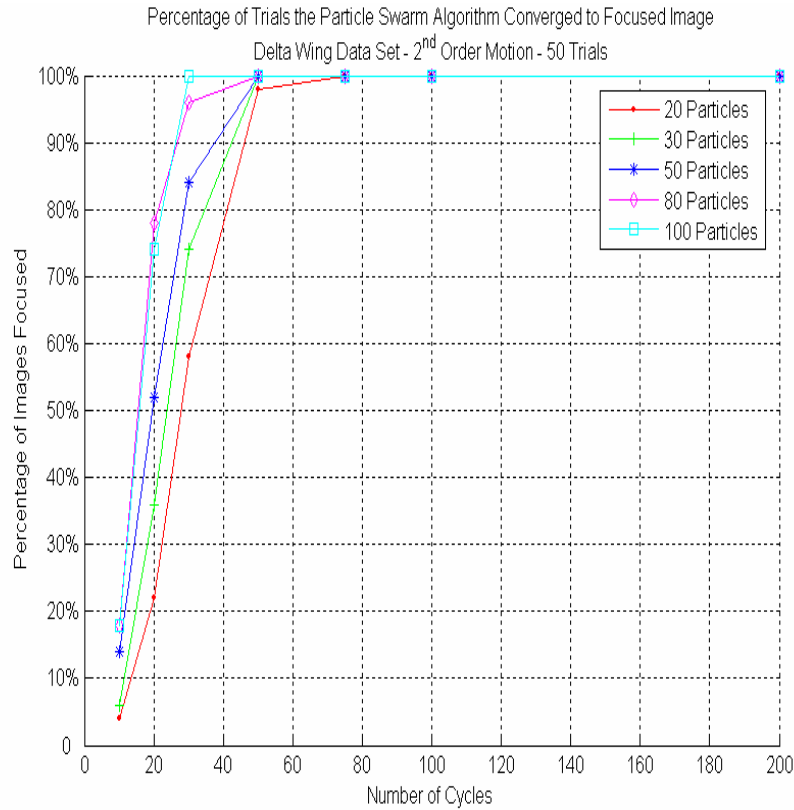


Figure 47. Percentage of Images Focused Using the Particle Swarm Algorithm as a Function of Swarm Size and Number of Cycles – Delta Wing Data Set.

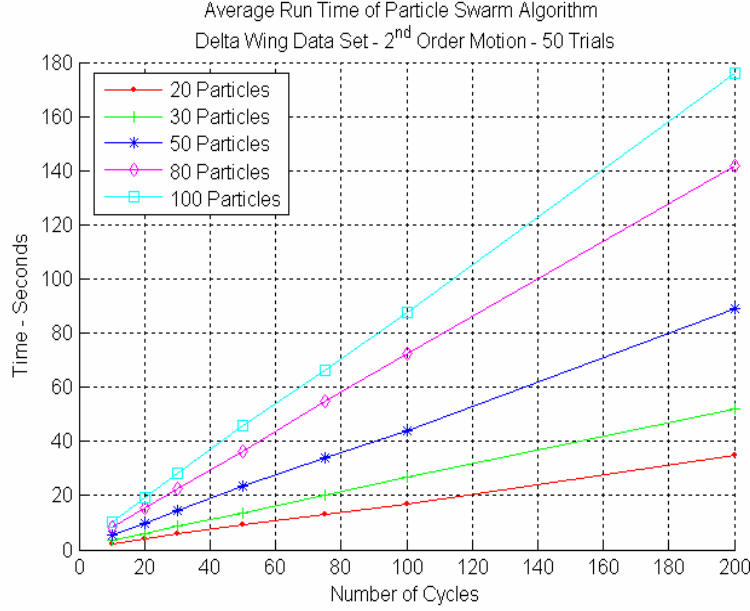


Figure 48. Average Run Time of the Particle Swarm Optimization Algorithm as a Function of Swarm Size and Number of Cycles – Delta Wing Data Set.

D. COMPARISON BETWEEN GA AND PSO ISAR FOCUSING ALGORITHMS

Comparisons between the GA and the PSO algorithms are made in [7] and [11] and they typically find that both evolutionary search techniques produce similar results. In particular, [11] states that during some optimization trials, which in this case was optimizing the complex weights of an antenna array, the GA outperforms the PSO while in others the PSO outperforms the GA. In the case of the ISAR focusing problem, for imaging intervals where the AJTF algorithm was a valid mathematical model, the resulting focused images, produced by the GA and PSO algorithms, were of equal quality as both search techniques were capable of finding the optimal search parameters. When a poorly focused image was produced by the GA or PSO, it was from an imaging interval of the 6 – Point Delta Wing experimental data set that possessed qualities that were in severe violation of the assumptions of the AJTF motion compensation model. These imaging intervals were of equivalent poor quality regardless of whether the GA or the PSO was used for the focusing parameter search. Also, by examining the average run time graphs of both the GA and the PSO, it can be concluded that, for equivalent calculations of the cost function, the two algorithms are equally fast at processing through the user-defined number of generations or cycles. However, if the convergence graphs for the B727 simulated

data set in Figures 28 and 41 and the convergence graphs for the 6 – Point Delta Wing experimental data set in Figures 34 and 47 are examined, it is shown that there is a significant difference in performance between the two search algorithms. Figure 49 combines part of the GA and the PSO results from their respective convergence graphs with the lines with triangles denoting GA results and lines with circles denoting PSO results. What can be seen from the graphs in Figure 49 is that the PSO written for this thesis consistently outperforms the GA by reliably converging to a focused image even when the PSO has less particles in the swarm than the GA has parents in its population. For the B727 data set, a PSO with 10 particles outperforms a GA with a population size of 10 and 16 and performs almost as well as a GA with a population size of 26. In the 6 – Point Delta Wing experimental data set, a PSO with 20 particles outperforms a GA with a population size of 20, 30 and 50. Also, from a programming point of view, a PSO is much easier to code and to troubleshoot and its search pattern is judged merely by Equation (38) and, although some of the constants given by [7] can be changed, there was no requirement to implement any parameter changes for the PSO in this thesis. For a GA, setting a correct mutation rate, reinsertion rate and selecting parents correctly for breeding are all of critical importance for reliable convergence and can take considerable effort to set at a correct level that balances global and local searches. For these reasons, the PSO is determined to be better suited for ISAR focusing and only the PSO will be used for the subsequent sections of this thesis.

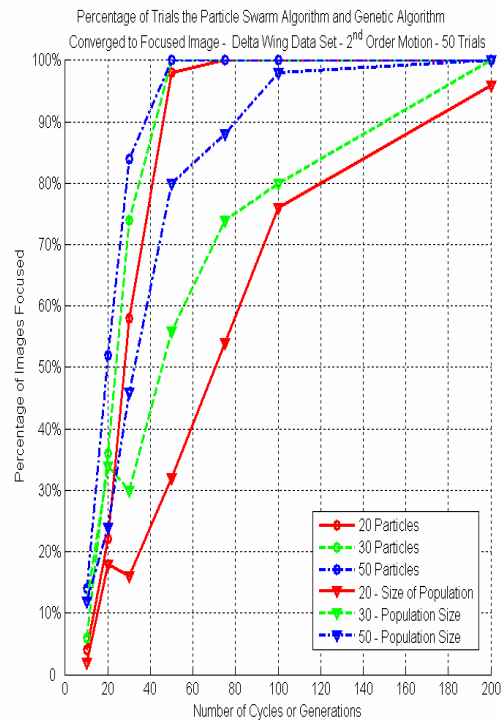
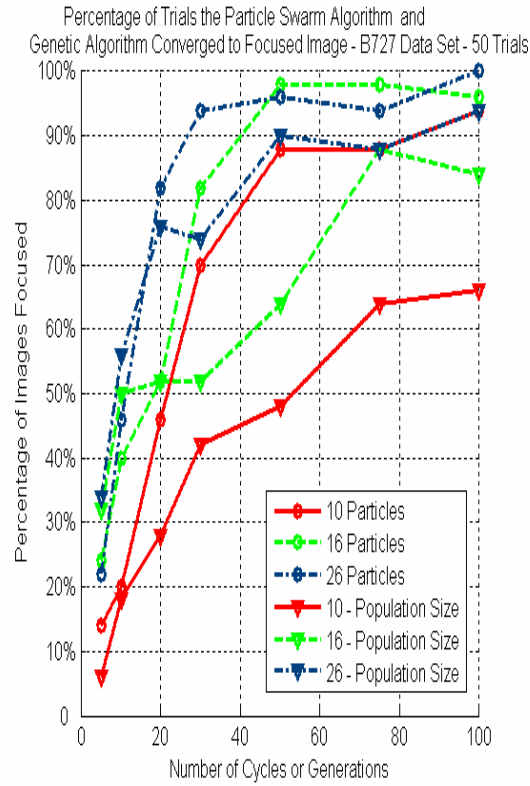


Figure 49. Combined GA (top) and PSO Convergence Graphs for Selected Population / Swarm Sizes.

E. HIGHER ORDER MOTION COMPENSATION

Higher order motion occurs when a target's motion is extremely chaotic. This creates the situation where searching for the 2nd-order search parameters, f_{12} and f_{22} , is insufficient to focus the ISAR image. This has been partially addressed already in the B727 simulated data set where it was required to find parameters $\{f_{12}, f_{13}\}$ and $\{f_{22}, f_{23}\}$ to focus the image. However, with only 256 pulses in the cross-range, which translates to an ideal solution space of 262,144 possible combinations, the PSO is able to quickly traverse the solution space and find the focusing parameters. The problem with higher-order motion compensation occurs when there are a large number of pulses in the cross-range. For example, for 3rd-order motion compensation, the solution space of the 2048 pulse 6 – Point Delta Wing experimental data set increases from 4096 possibilities to 16.7×10^6 possibilities. If 4th-order motion compensation is required, the solution space expands to 68.7×10^9 possibilities. Also, do not forget, that these solution spaces must always be traversed twice, once to extract translational and the other to extract rotational motion compensation parameters. This creates an enormous search space that cannot be quickly traversed by even the most efficient PSO.

Imaging interval 13 of the 6 – Point Delta Wing experimental data set is given as an example and its unfocused image is shown in Figure 50. This image possesses 4th-order motion error and cannot be focused with 2nd-order motion compensation. The existence of higher-order motion can be seen by examining a range bin in the time-frequency plane. As explained in [3], scatterers with higher-order motion will be curved as compared to 2nd-order motion error whose scatterers are tilted in the time-frequency plane. The time-frequency transforms of the scatterers in the unfocused range bins 25 and 19 are shown on the left of Figure 51. The fact that they are curved is a strong indication of the presence of higher-order motion error. If the standard PSO AJTF focusing algorithm, using 4th-order motion compensation, is applied to the image, an extremely well-focused image is produced as seen in Figure 52. Also, the right side of Figure 51 shows that the time-frequency transforms of the scatterers from the focused image have been straightened since the time-varying Doppler has been removed from the signal. Unfortunately, this requires approximately 1×10^6 calculations of the cost function. This limits the PSO

usefulness since this requires an excessive run time of 1 hour and 10 minutes. The way around this is to adapt the AJTF PSO algorithm to perform the search in the same pattern as the exhaustive search, which was introduced earlier in this Chapter. In this variation and, like in the exhaustive search, the PSO is applied to searching for f_2 . Once the optimal fit is found, f_3 is searched for with the PSO. Once the optimal f_3 is found, as long as the addition of the f_3 parameter adds to the fitness value as calculated by Equation (28), it is added to the basis function. If the fitness decreases, f_3 is set equal to zero. This process is continued for as many degrees of motion error as required and is completed for both translational and rotational motion compensation. The resulting focused image, for 2nd, 3rd and 4th-order motion compensation is shown in Figure 53. The 4th-order motion compensated image is shown on the left and though it is not as good as the image in Figure 52, it can be achieved in 20 seconds with only 4500 calculations of the cost function. Therefore, a sub-optimal, but usable, image can be achieved from an image blurred by higher-order motion using this focusing method which is exceptionally fast even with a large number of pulses in the cross-range.

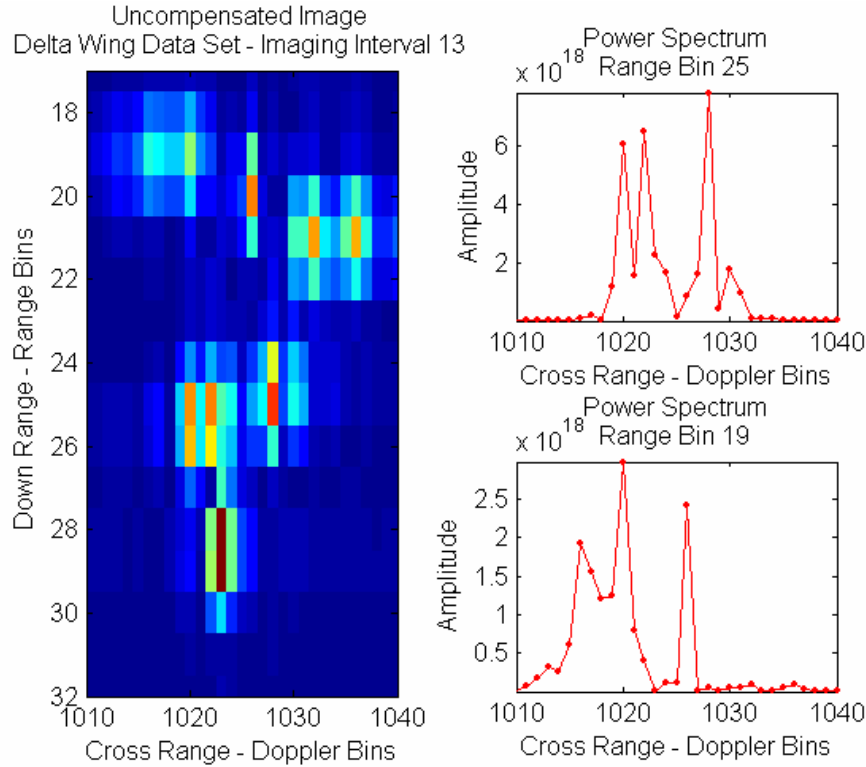


Figure 50. Unfocused Imaging Interval 13.

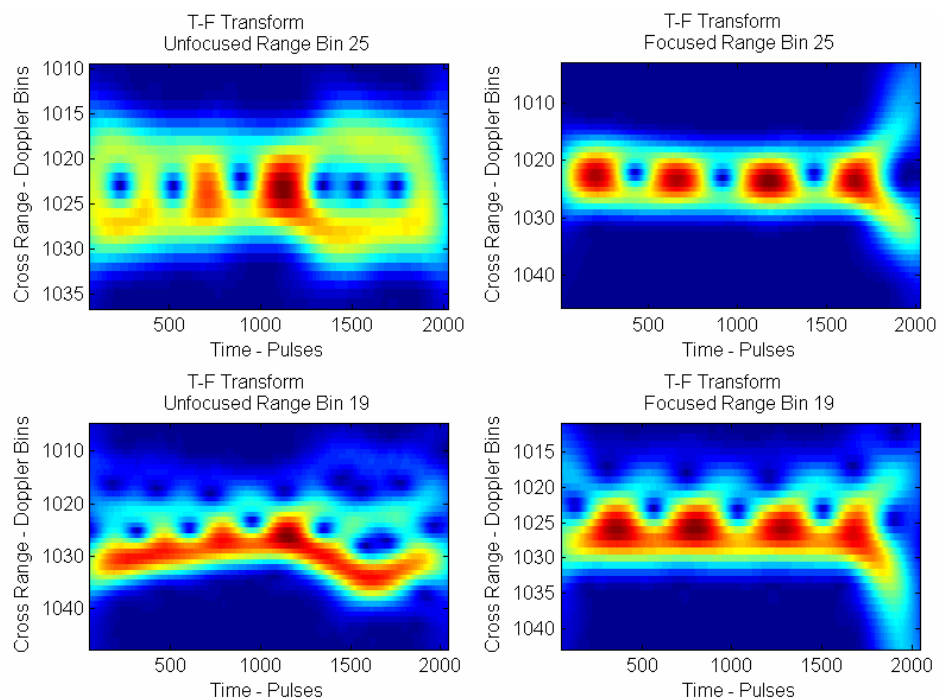


Figure 51. Time-Frequency Transforms of Range Bins 25 and 19 – Left: Unfocused
Right: Focused.

Final Image after Rotational Motion Compensation
Delta Wing Data Set - Imaging Interval 13

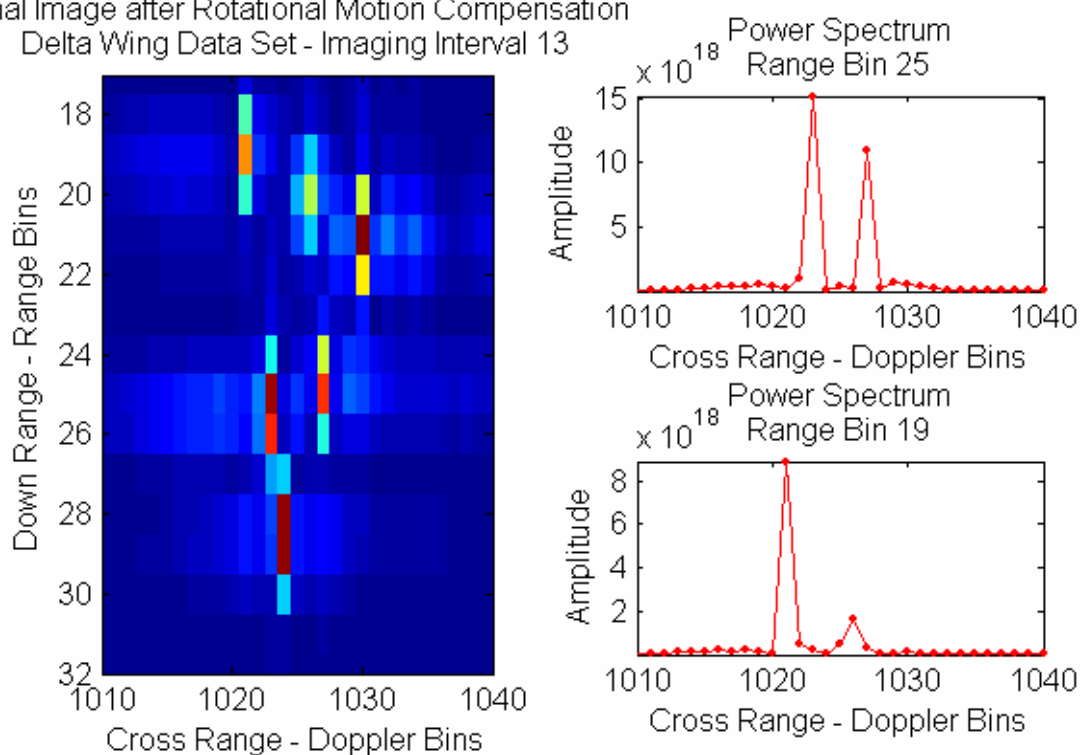


Figure 52. Focused Imaging Interval 13.

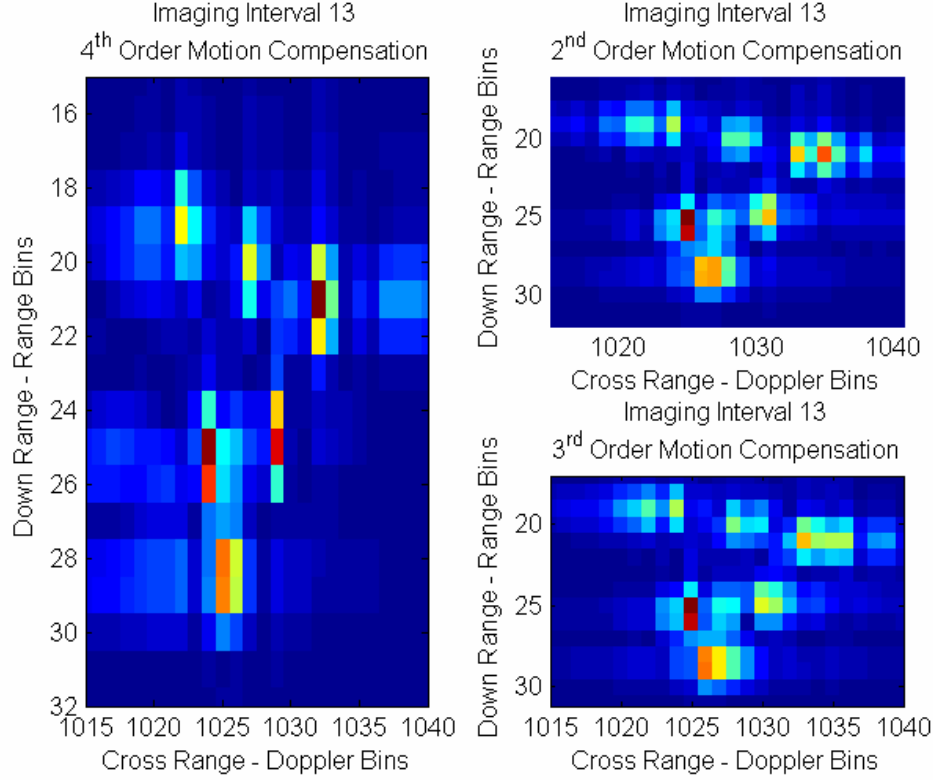


Figure 53. Image Interval 13 with 2nd, 3rd and 4th-Degree Motion Compensation, Reduced Solution Space.

F. CHAPTER SUMMARY

In this chapter, we have examined the construct of the solution space in the ISAR focusing problem and saw that it possessed many local maxima but only one global maximum. This characteristic makes the solution space ill-suited for conventional searches but well suited for the GA and the PSO evolutionary searches. It was also shown that the GA and the PSO provided considerable computational time saving over the exhaustive search which is the method normally used for parameter extraction in the AJTF algorithm. Also, it was shown that the PSO performed considerably better than the GA in the ISAR focusing task. Finally, higher-order motion compensation, using the PSO AJTF, was demonstrated on an imaging interval of the 6 – Point Delta Wing experimental data set. In the next chapter, we will examine what occurs when 3D motion enters the radar signal and how this can be accommodated since this is a violation of a fundamental assumption of the AJTF.

V. 3D MOTION DETECTION

In Chapter IV, a GA and PSO algorithm were successfully adapted to the optimization of the AJTF algorithm which provided a solution to the computational intensity required for successful ISAR focusing that was one of the algorithms major weaknesses. A number of imaging intervals, which are now well focused in the cross-range, were presented. These imaging intervals, however, follow the mathematical model of the AJTF. The next step in this thesis is to examine an effective method for dealing with imaging intervals which violate the AJTF's mathematical model.

A. INTRODUCTION TO THE 3D MOTION MODEL AND THE DETECTION METHOD

One of the basic assumptions of the adaptive joint time-frequency motion compensation model is that the rotational motion used to form the ISAR image is confined to a 2D plane during the coherent processing interval, T_{CPI} . When motion in the roll plane of the aircraft occurs during the coherent processing interval, as shown in Figure 54, 3D motion is present in the ISAR return signal and the motion error cannot be compensated by the motion compensation model in its current form. Due to the difficulty in compensating for 3D motion error, Chen, Li and Ling, in [3] and [4], developed the linearity of phases method to detect the presence of 3D motion. Thayaparan further expanded on the 3D detection method in [12] and it is their work that was adapted to the AJTF PSO motion compensation algorithm for the purposes of 3D motion detection in this thesis. The goal of 3D motion detection is to determine which imaging intervals of an ISAR data set contains a low degree of 3D motion and can be focused using the AJTF algorithm. Imaging intervals found to possess a high degree of 3D motion are simply discarded and not presented to the ISAR operator.

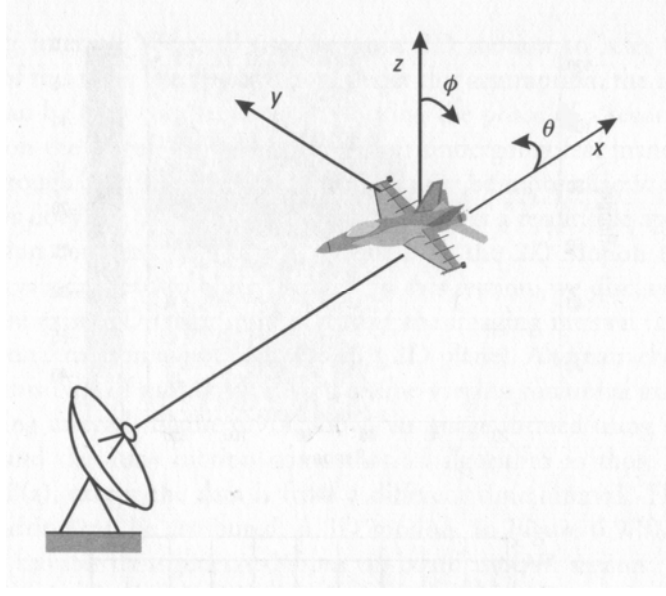


Figure 54. Engagement of a Target with 3D Motion (From [3].)

Now that there is 3D motion in the ISAR signal, Equation (1) can be written in a more general form of [3]:

$$s(x, t) = \sum_{k=1}^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [R(t) + x_k + y_k \theta(t) + z_k \phi(t)] \right\} \quad (42)$$

where $\phi(t)$ describes the independent angular parameter in the roll plane and (x_k, y_k, z_k) are the 3D coordinate position of the k^{th} scatterer. Chen explains in [3] that Equation (42) will reduce to the 2D model in two cases. The first case is when z_k is small and the $\phi(t)$ phase term can be neglected. The second case is when there exists a linear relationship between $\theta(t)$ and $\phi(t)$ such that $\phi(t) = a\theta(t)$ [3].

The first step towards detecting 3D motion is to remove translation motion error from the signal by finding the appropriate basis function, $h_p(t)$, as previously described and multiplying the signal by its conjugate. With the translational motion error removed, Equation (42) reduces to [4]:

$$\begin{aligned}
s(x, t)_T &= \sum_{k=1}^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [\Delta x_k + \Delta y_k \theta(t) + \Delta z_k \phi(t)] \right\} \\
&= \sum_{k=1}^{N_k} A_k \exp \left\{ -j \frac{4\pi f_0}{c} [\Delta x_k + (\Delta y_k + a \Delta z_k) \theta(t)] \right\}.
\end{aligned} \tag{43}$$

Now that the translational motion error has been compensated, it becomes necessary to find the rotational motion compensation search parameters, $\{f_{21}, f_{22}, f_{23}, \dots\}$, that are appropriate to formulate the phase function, $\Theta_R(t)$, as in Equation (20), for several different point scatterers in the target. For the purposes of this thesis, 3 point scatterers from the 6 – point Delta Wing data set are used and the AJTF PSO that was developed above is adapted to extract the search parameters.

Once all three phase functions have been extracted we have:

$$\begin{aligned}
\Theta_{R1}(t) &= f_{21_1} t + \frac{1}{2!} f_{22_1} t^2 + \frac{1}{3!} f_{23_1} t^3 + \dots \\
\Theta_{R2}(t) &= f_{21_2} t + \frac{1}{2!} f_{22_2} t^2 + \frac{1}{3!} f_{23_2} t^3 + \dots \\
\Theta_{R3}(t) &= f_{21_3} t + \frac{1}{2!} f_{22_3} t^2 + \frac{1}{3!} f_{23_3} t^3 + \dots
\end{aligned} \tag{44}$$

For signals that do not possess 3D motion, any of the above phase functions, $\Theta_{R1}(t)$, $\Theta_{R2}(t)$ or $\Theta_{R3}(t)$, would be suitable for rotational motion compensation which would create an ISAR image that is well focused in the cross-range. For 3D motion detection, Thayaparan's work in [12] is followed and an average phase function is formed:

$$\Theta_{avg}(t) = \left(\frac{f_{21_1} + f_{21_2} + f_{21_3}}{3} \right) t + \frac{1}{2!} \left(\frac{f_{22_1} + f_{22_2} + f_{22_3}}{3} \right) t^2 + \frac{1}{3!} \left(\frac{f_{23_1} + f_{23_2} + f_{23_3}}{3} \right) t^3 + \dots \tag{45}$$

The purpose to formulating $\Theta_{avg}(t)$ is that as more and more search parameters are extracted from more and more point scatterers suitable for rotational motion compensation, $\Theta_{avg}(t) \xrightarrow{\text{approaches}} \Theta_{ideal}(t)$ where $\Theta_{ideal}(t)$ is the best phase function possible that describes the rotational motion error in the whole signal and, after rotational motion compensation, produces the best image. In this discussion, the number of point scatterers is

limited to three since the maximum number of scatterers in the 6 – Point Delta Wing data set is six and not all scatterers are suitable for rotational motion compensation.

Now, if the imaging interval possesses a low degree of non-linearity, the plot of Θ_{R1} , Θ_{R2} or Θ_{R3} as a function of Θ_{ideal} would be a straight line as in Figure 55. Therefore, the first step in determining the degree of non-linearity is to find the straight line of best fit, in a least-squares sense using the MatLab *polyfit* function, between Θ_{R1} , Θ_{R2} or Θ_{R3} and Θ_{ideal} . The second step is to plot Θ_{R1} , Θ_{R2} or Θ_{R3} as a function of Θ_{ideal} and measure the percentage that this plot deviates from the line of least-squares fit using:

$$NF_1 = \sum_{t=1}^N \left(\frac{|\Theta_{R1}(\Theta_{ideal}(t)) - \text{LSF}\{\Theta_{R1}(\Theta_{ideal}(t))\}|}{\text{LSF}\{\Theta_{R1}(\Theta_{ideal}(t))\}} \right) \quad (46)$$

where NF_1 is the degree of non-linearity between $\Theta_{R1}(t)$ and $\Theta_{ideal}(t)$, $\Theta_{R1}(\Theta_{ideal}(t))$ is the plot of $\Theta_{R1}(t)$ as a function of $\Theta_{ideal}(t)$, $\text{LSF}\{\Theta_{R1}(\Theta_{ideal}(t))\}$ is the straight line least-squares fit between $\Theta_{R1}(t)$ and $\Theta_{ideal}(t)$ and N is the number of pulses in the cross-range. This procedure is repeated to generate NF_2 and NF_3 for the other two phase functions and the imaging interval is assigned its degree of non-linearity by simply averaging the non-linearity functions [12]:

$$NF_{Interval} = \frac{NF_1 + NF_2 + NF_3}{3}. \quad (47)$$

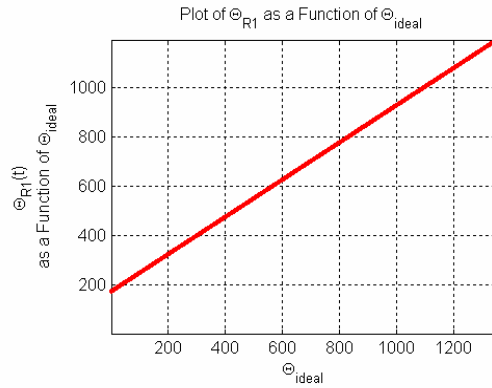


Figure 55. Plot of Θ_{R1} as a Function of Θ_{ideal} when no 3D motion is present (After [12].)

Figure 56 shows the Non-Linearity of Phases plot for an imaging interval with a high degree of non-linearity. Note that the plot of $\Theta_{R1}(\Theta_{ideal}(t))$ deviates significantly from the line of least-squares fit. Figure 57 shows the Non-Linearity of Phases plot for an imaging interval with a very low degree of non-linearity. Note that the plot of $\Theta_{R1}(\Theta_{ideal}(t))$ in this case is plotted exactly on top of the line of least-squares fit.

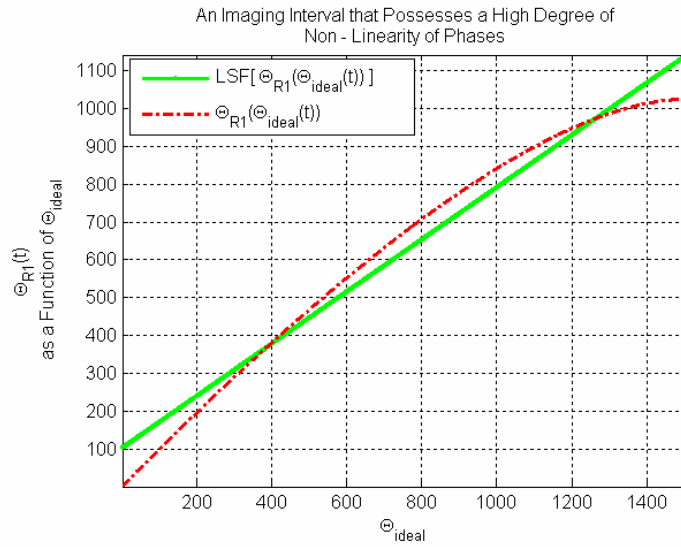


Figure 56. Imaging Interval Possessing a High Degree of Non-Linearity (After [12].)

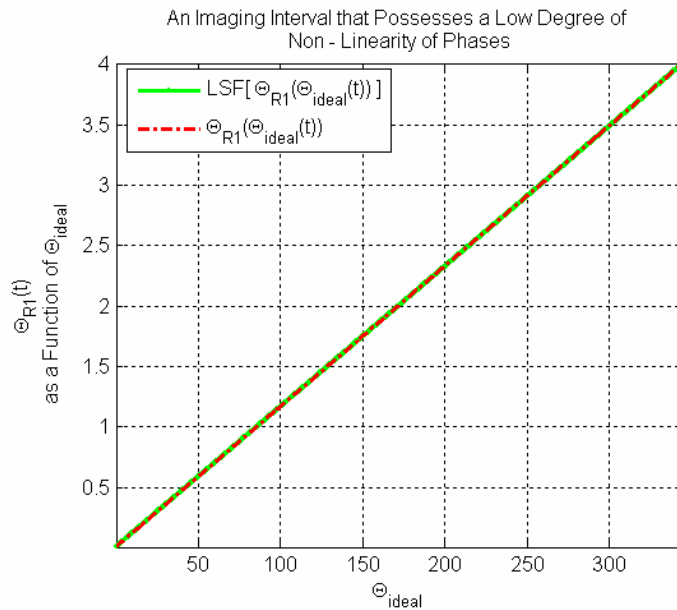


Figure 57. Imaging Interval Possessing a Low Degree of Non-Linearity (After [12].)

B. 3D MOTION DETECTION RESULTS

The 3D motion detection algorithm was tested against the 6 – Point Delta Wing data set with 2048 and 1024 pulses in the cross-range and 2nd and 4th-degree motion compensation. The primary goal of this section is to demonstrate the capability of Thaya-paran's 3D motion detection algorithm to differentiate between imaging intervals possessing a high degree of non-linearity and those that possess a low degree of non-linearity after the algorithm has been adapted to the AJTF PSO algorithm,. Secondly, the effect of choosing different sizes of imaging intervals and different degrees of motion compensation in detecting 3D motion will be examined.

1. Imaging Intervals with 2048 Pulses and 2nd-Degree Motion Compensation

If the 3D motion detection algorithm is applied to the 6 – Point Delta Wing experimental data set with its imaging intervals divided into 2048 pulses and 2nd-degree motion compensation is applied, the non-linearity of phases graph is shown in Figure 58.

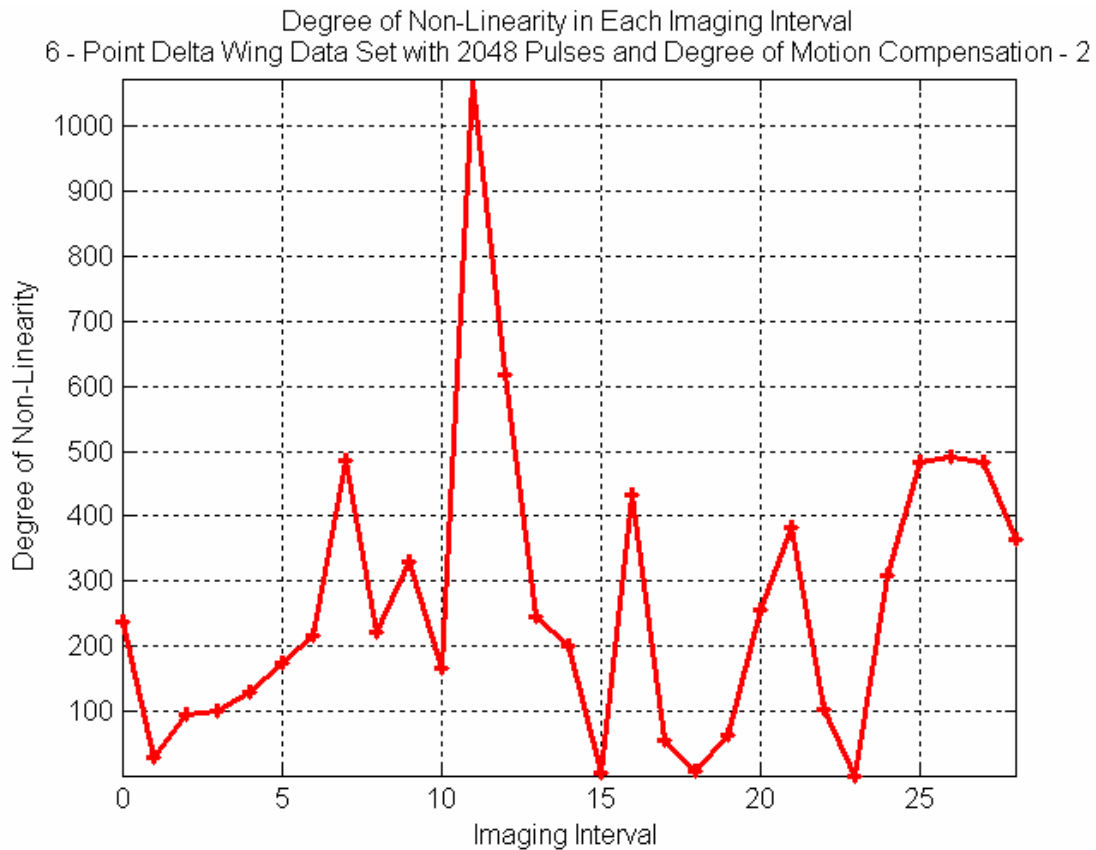


Figure 58. Degree of Non-Linearity of Imaging Intervals of 2048 Pulses and 2nd-Degree Motion Compensation.

Figure 58 indicates that imaging intervals with a low degree of non-linearity of phases are imaging intervals 23, 15, and 18. If these images are focused using the AJTF PSO designed for this thesis, the resulting well focused images are shown in Figures 59 – 61:

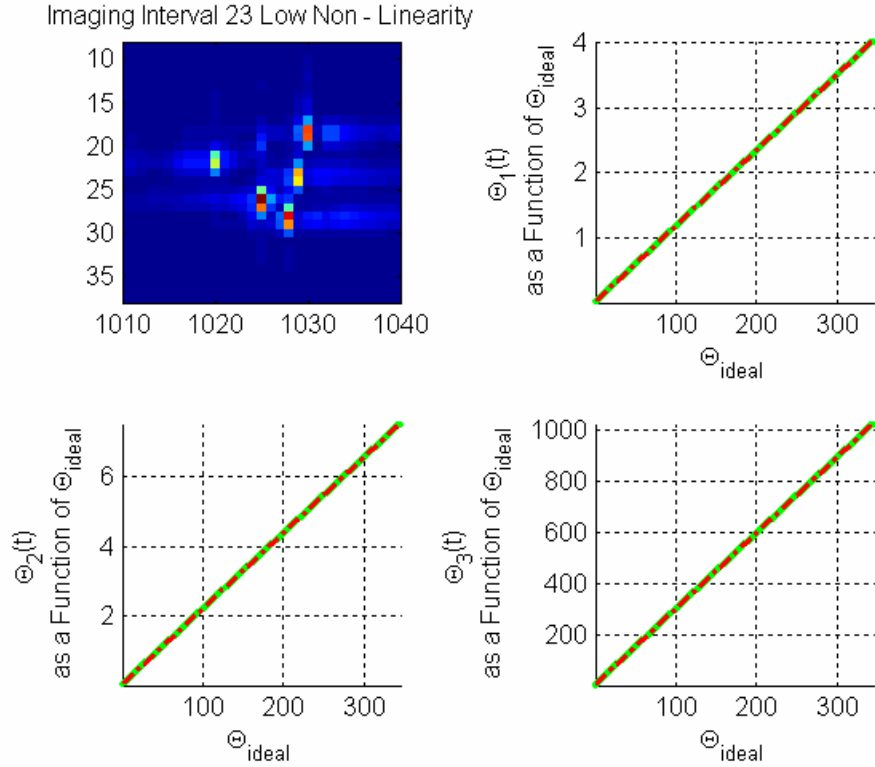


Figure 59. Imaging Interval 23 – Well Focused with Low Degree of Non-Linearity of Phases.

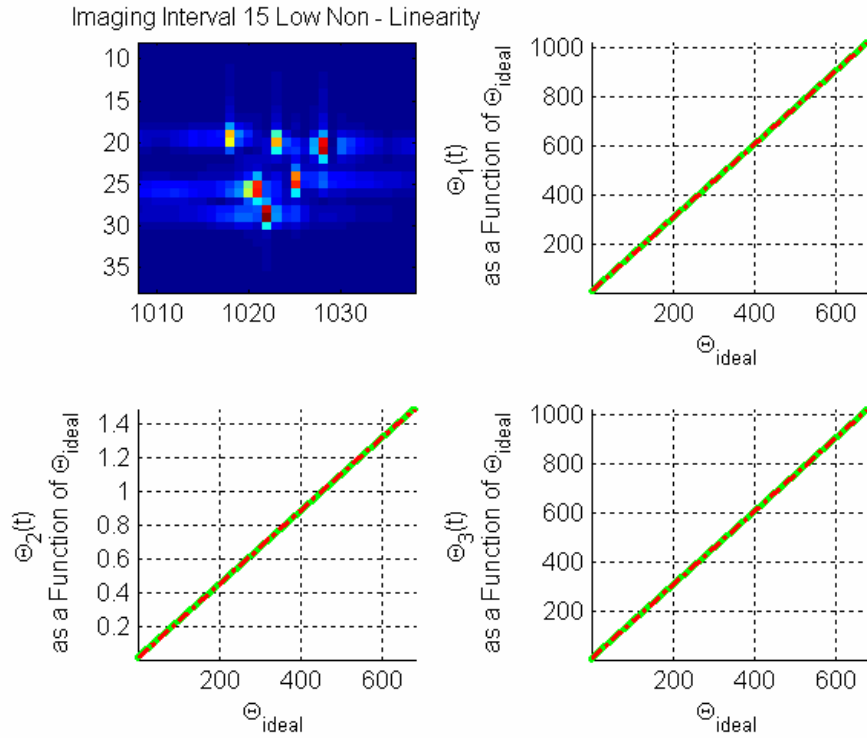


Figure 60. Imaging Interval 15 – Well Focused with Low Degree of Non-Linearity of Phases.

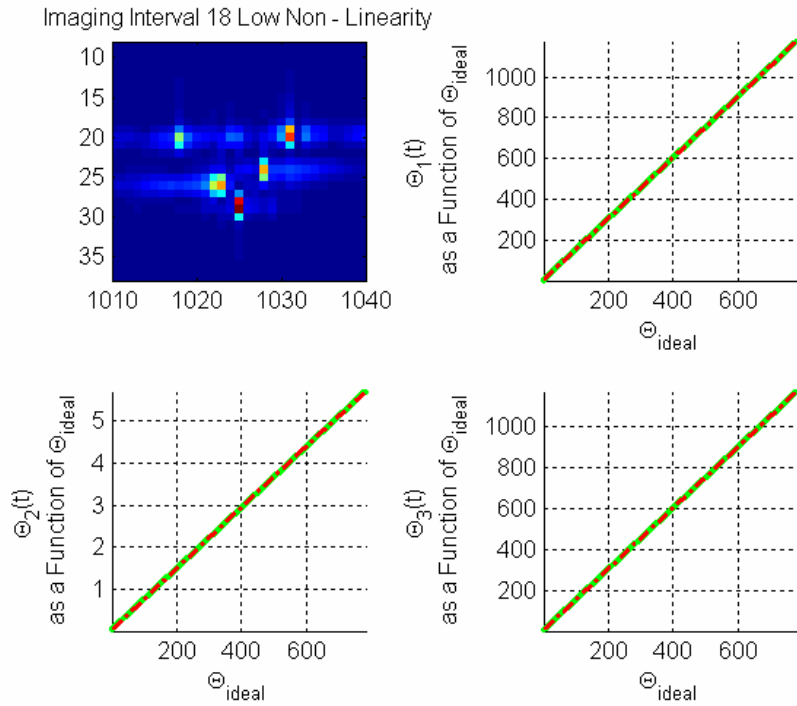


Figure 61. Imaging Interval 18 – Well Focused with Low Degree of Non-Linearity of Phases.

Figure 58 also indicates that imaging intervals 11, 12 and 26 possess a high degree of non-linearity of phases. If these images are focused using the AJTF PSO designed for this thesis, the following poorly focused images, shown in Figures 62 - 64 result:

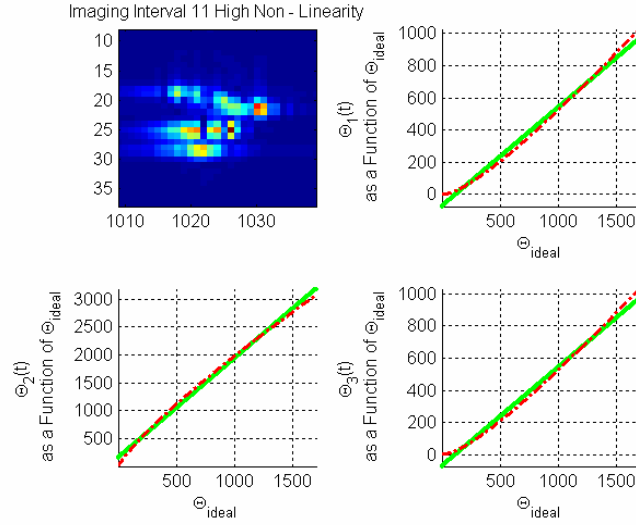


Figure 62. Imaging Interval 11 – Poorly Focused with High Degree of Non-Linearity of Phases.

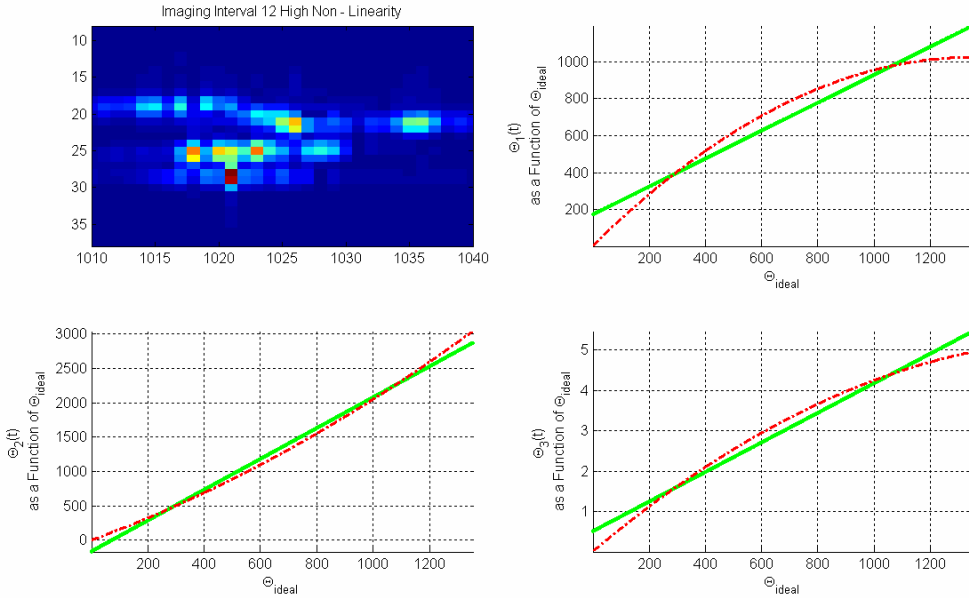


Figure 63. Imaging Interval 12 – Poorly Focused with High Degree of Non-Linearity of Phases.

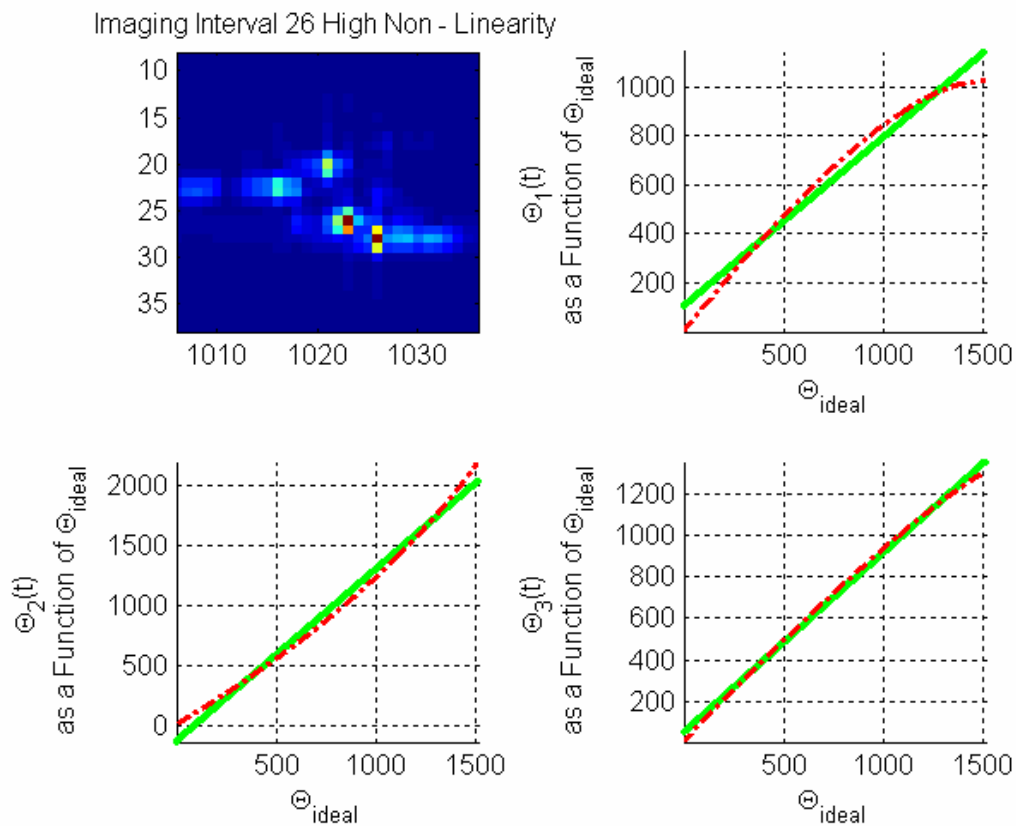


Figure 64. Imaging Interval 26 – Poorly Focused with High Degree of Non-Linearity of Phases.

2. Imaging Intervals with 2048 Pulses and 4th-Degree Motion Compensation

Now, if 4th-degree motion compensation (i.e., motion compensation parameters $\{f_{21}, f_{22}, f_{23}, f_{24}\}$) are used, the following non-linearity of phases graph is generated and shown in Figure 65:

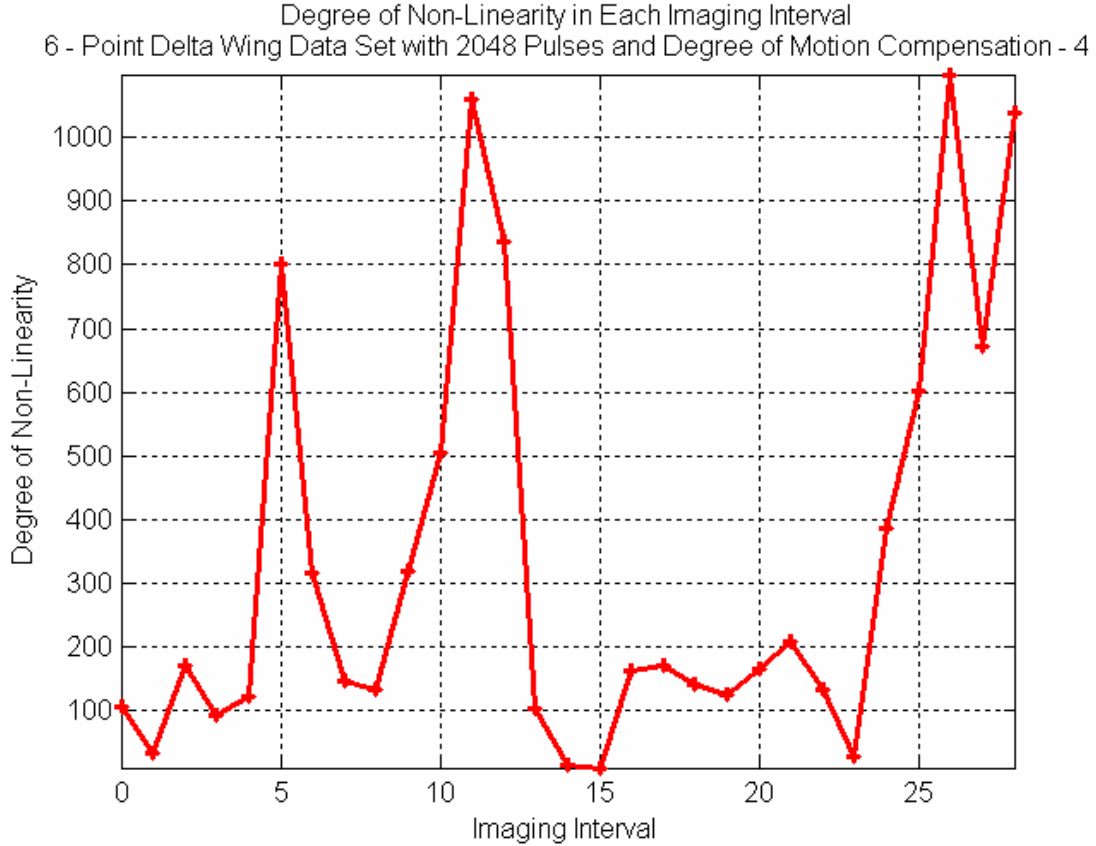


Figure 65. Degree of Non-Linearity of Imaging Intervals of 2048 Pulses and 4th-Degree Motion Compensation.

In general, there is consistency in that those imaging intervals which had a low degree of non-linearity of phases in Figure 58, still register as having a low degree of non-linearity in Figure 65. The same applies for the imaging intervals with a high degree of non-linearity. There are two reasons for the differences between Figures 58 and 65. First, the AJTF PSO used to extract the search parameters has a certain randomness associated with its search pattern and it is possible that, for certain imaging intervals, incorrect search parameters were extracted and thus the degree of non-linearity calculation was

slightly off. Secondly, 4th-degree motion compensation is more accurate so, provided that all of the search parameters were extracted correctly, it provides a better measure on the degree of non-linearity.

Figure 65 indicates that the best three imaging intervals are imaging intervals 15, 14 and 23. Imaging intervals 15 and 23 are the same as before and are shown above. Imaging interval 14 is new and is now determined to be one of the best three imaging intervals in the data set. If this image in Figure 66 is compared to imaging interval 18 in Figure 61, it can be seen that imaging interval 14 is a better image than the one produced by imaging interval 18.

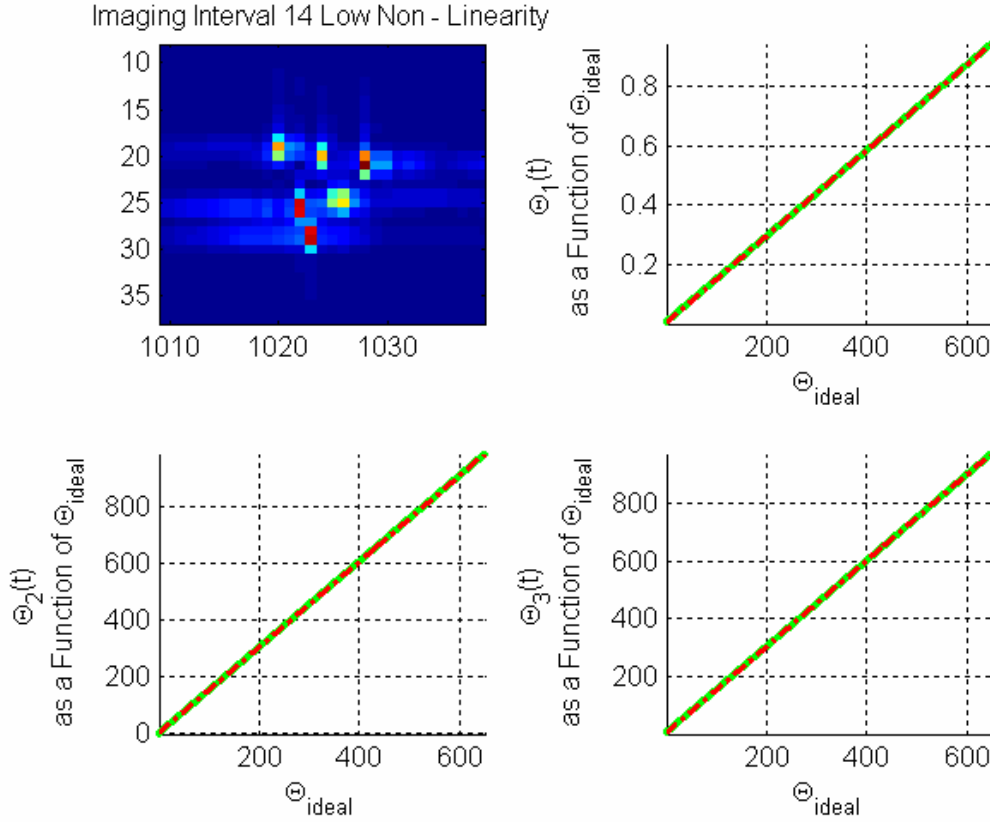


Figure 66. Imaging Interval 14 – Well Focused with Low Degree of Non-Linearity of Phases.

Figure 65 indicates that the three worst imaging intervals are imaging intervals 26, 11 and 28. Imaging interval 12, which is a particularly bad imaging interval and has been shown before in Figure 63, falls off the list of the top 3 worst imaging intervals.

Fortunately, it is still ranked as having a high degree of non-linearity and would still not be imaged. The new imaging interval, 28, is shown in Figure 67 and is not very well focused in the cross-range.

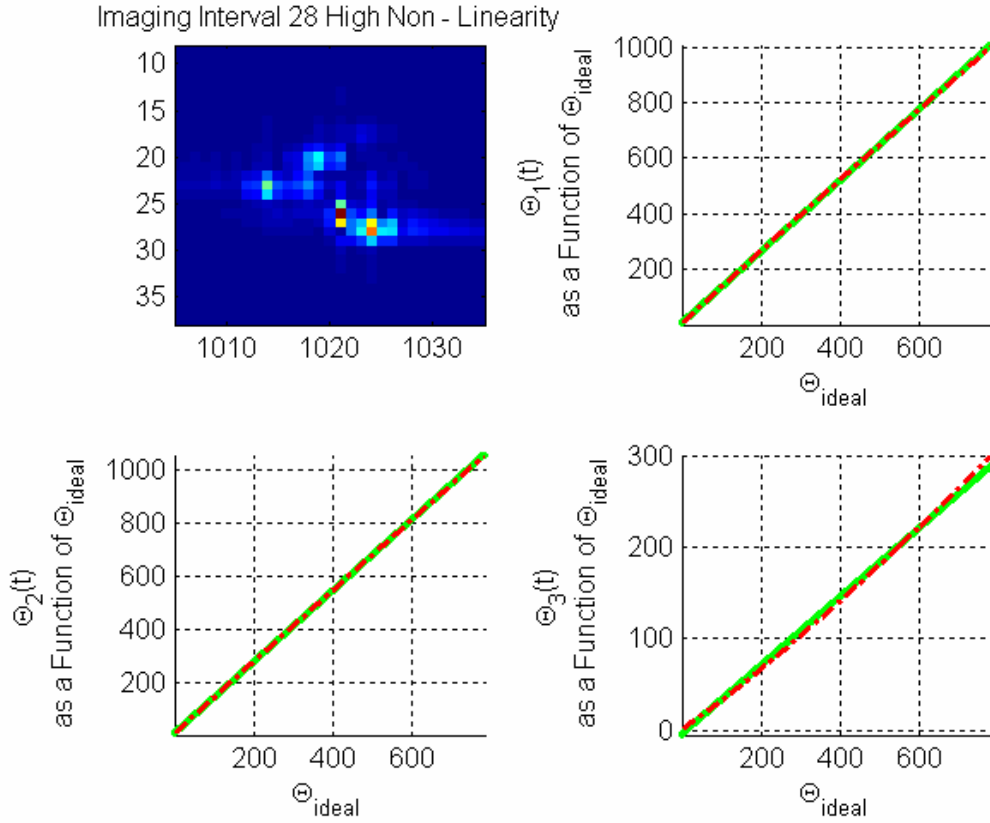


Figure 67. Imaging Interval 28 – Poorly Focused with High Degree of Non-Linearity of Phases.

3. Imaging Intervals with 1024 Pulses and 2nd-Degree Motion Compensation

There are several reasons to use imaging intervals with fewer pulses in the cross-range. The most obvious is that is that the T_{CPI} is smaller and subsequently there should be less motion error in each imaging interval. Secondly, since there are more imaging intervals available for processing, if an imaging interval has to be discarded, there are still many more imaging intervals available to work with. The downside to this is that with less pulses in the cross-range, the cross-range resolution, Δr_{cr} , will be less.

Figure 68 shows the non-linearity of phase graph for 1024 pulses in the cross-range and 2nd-degree motion compensation. The three best imaging intervals are imaging interval 45, 33 and 35 and are shown below in Figures 69 – 71. The three worst imaging intervals are imaging intervals 56, 25 and 29 and are shown in Figures 72 – 74. Again, imaging intervals with a low degree of non-linearity are easily focused while imaging intervals with high degree of non-linearity do not focus.

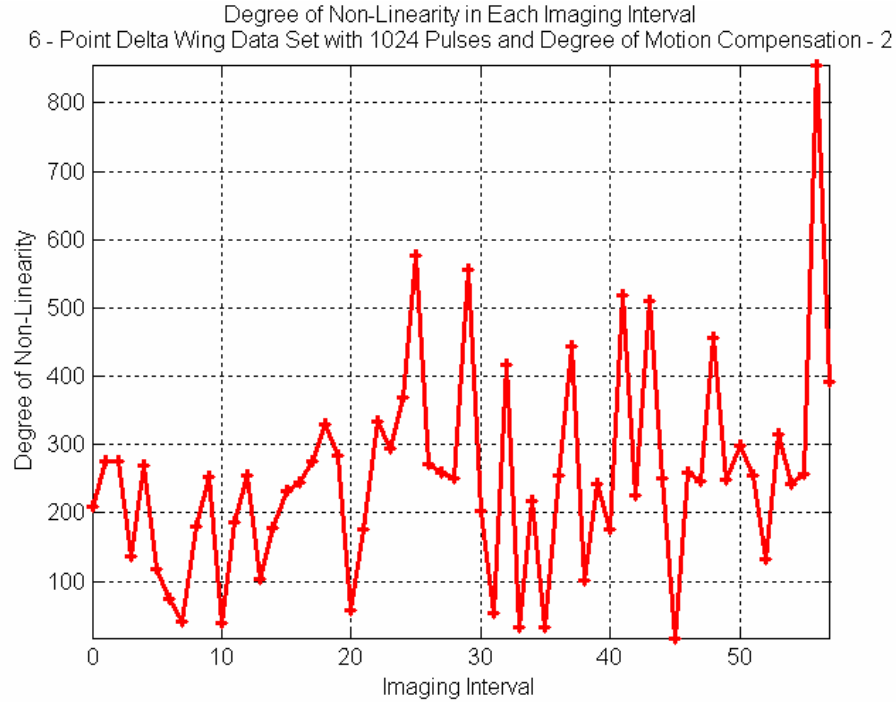


Figure 68. Degree of Non-Linearity of Imaging Intervals of 1024 Pulses and 2nd-Degree Motion Compensation.

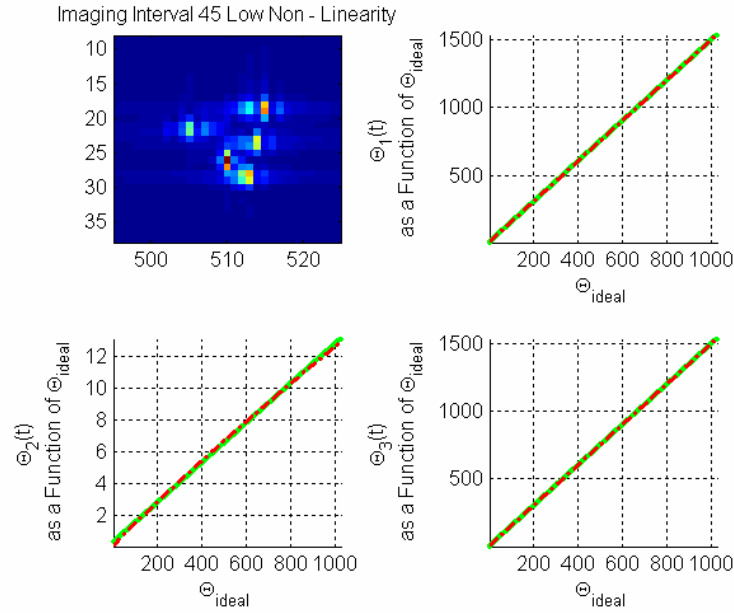


Figure 69. Imaging Interval 45 – Well Focused with Low Degree of Non-Linearity of Phases.

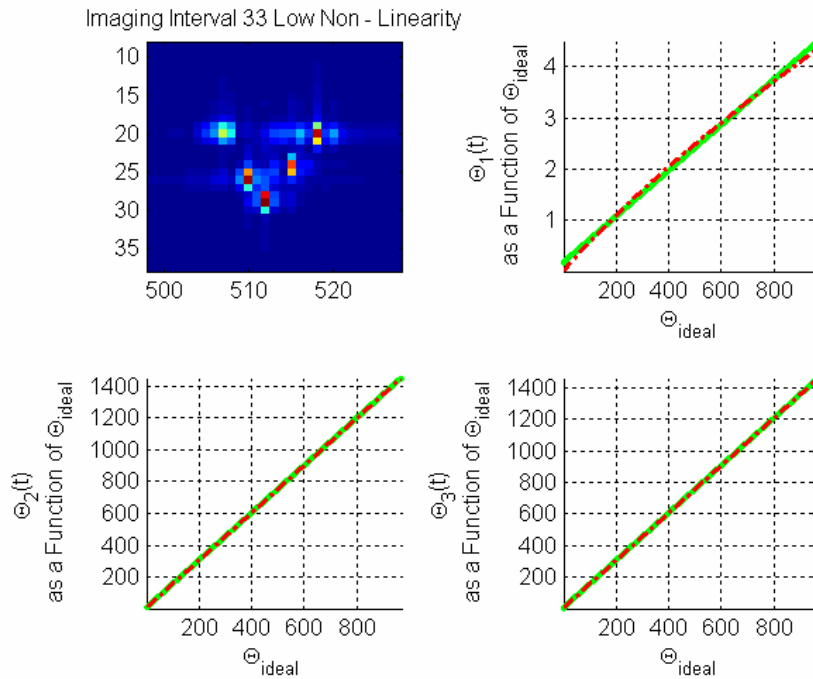


Figure 70. Imaging Interval 33 – Well Focused with Low Degree of Non-Linearity of Phases.

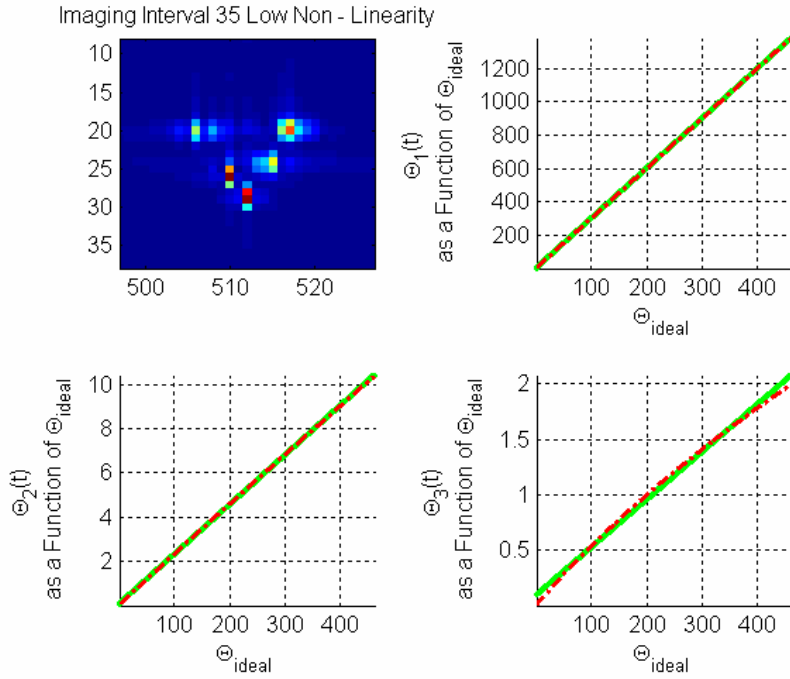


Figure 71. Imaging Interval 35 – Well Focused with Low Degree of Non-Linearity of Phases.

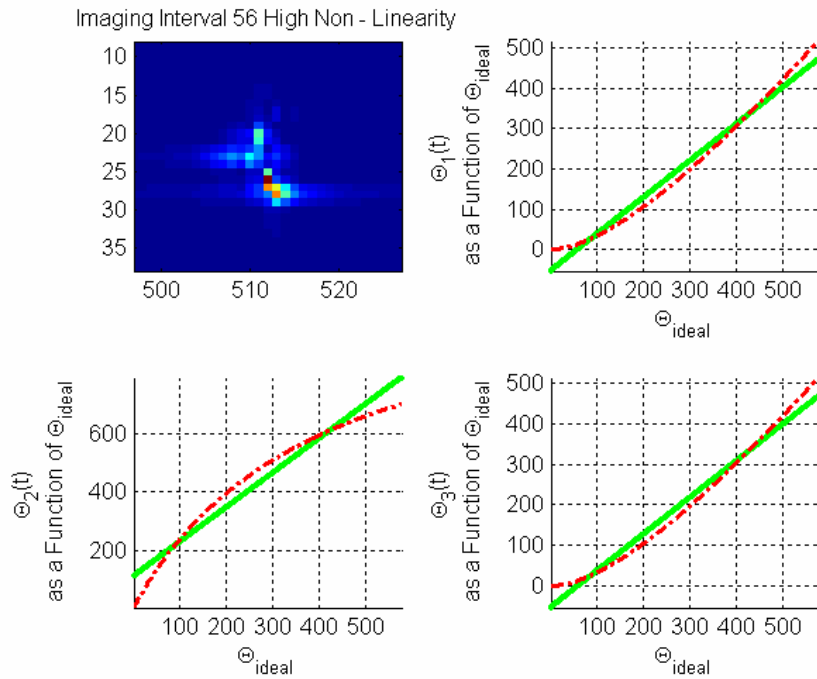


Figure 72. Imaging Interval 56 – Poorly Focused with High Degree of Non-Linearity of Phases.

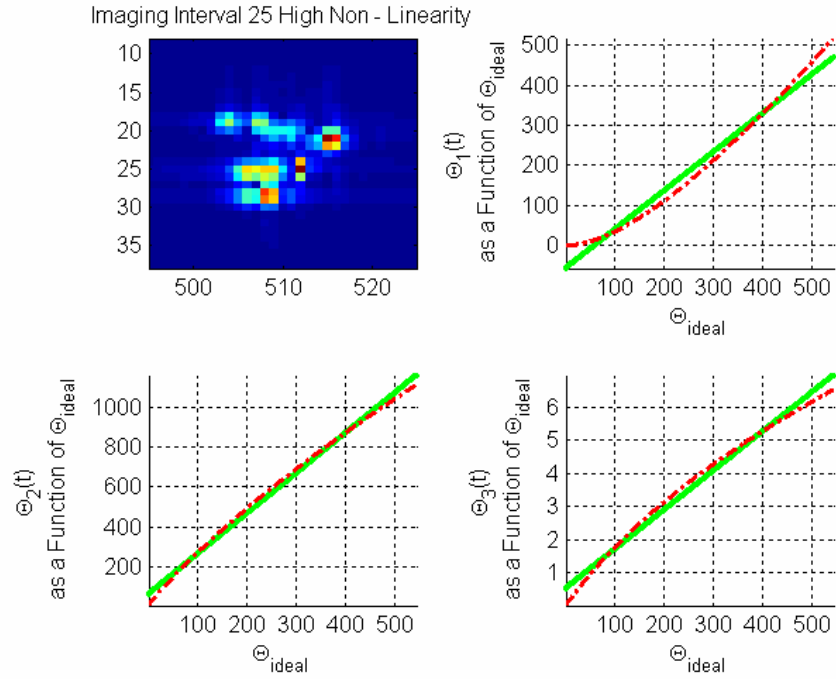


Figure 73. Imaging Interval 25 – Poorly Focused with High Degree of Non-Linearity of Phases.

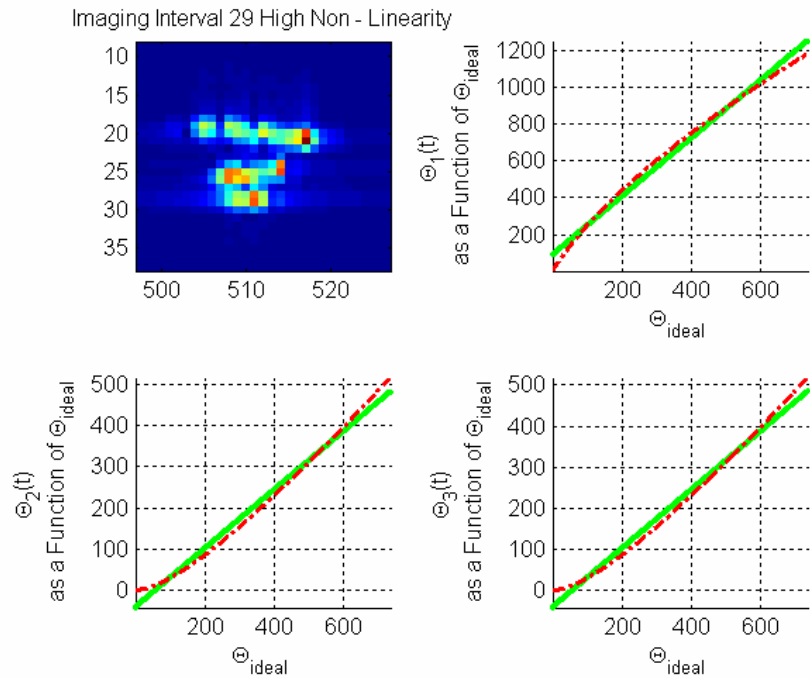


Figure 74. Imaging Interval 29 – Poorly Focused with High Degree of Non-Linearity of Phases.

4. Imaging Intervals with 1024 Pulses and 4th-Degree Motion Compensation

If 3D motion detection is repeated for 1024 pulse imaging intervals but this time using 4th-degree motion compensation, the degree on non-linearity graph is generated as in Figure 75. It can be seen from this graph that imaging intervals 10, 33 and 2 possess a low degree of non-linearity while imaging intervals 24, 56 and 29 possess a high degree of non-linearity. Good imaging intervals 10 and 2 are shown below in Figures 76 and 77 and poor imaging interval 24 is shown in Figure 78. Again, there is consistency between the 2nd and 4th-degree linearity of phase graphs (in Figures 68 and 75). Imaging intervals that register as having a high degree of non-linearity of phases for the 2nd-degree motion compensation also register as having a high degree of non-linearity of phases for 4th-degree motion compensation.

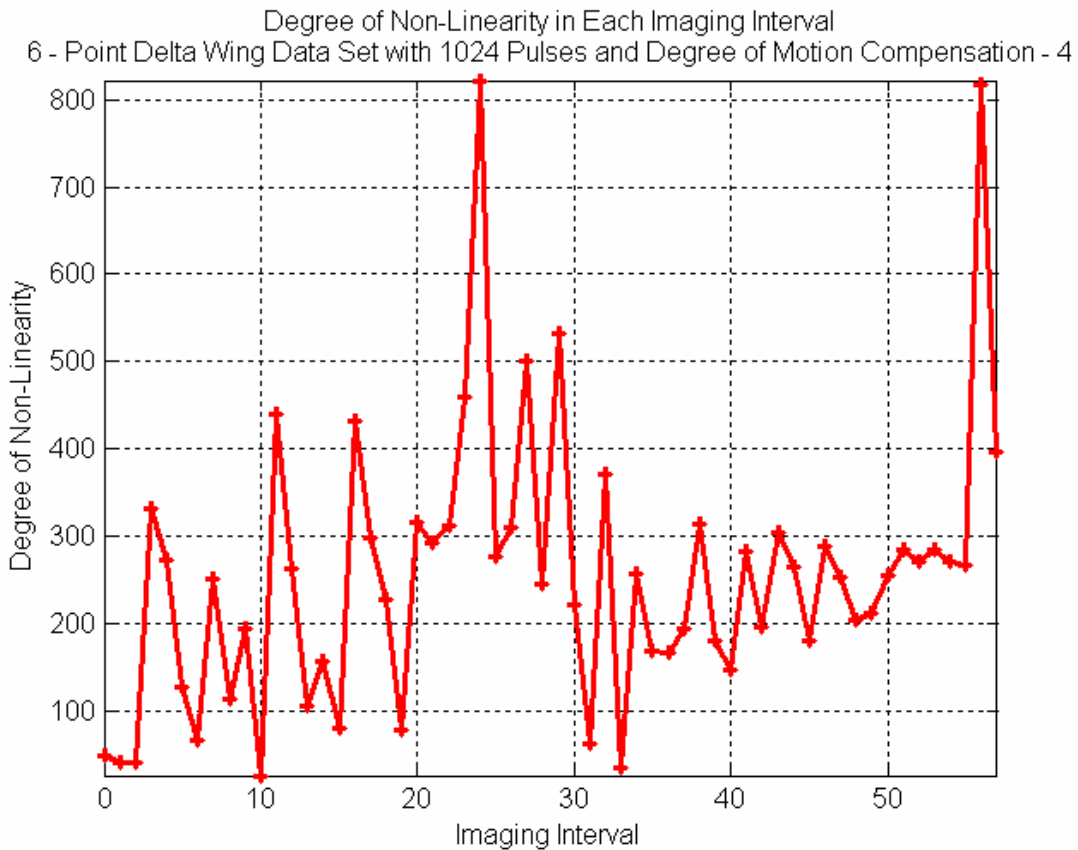


Figure 75. Degree of Non-Linearity of Imaging Intervals of 1024 Pulses and 4th-Degree Motion Compensation.

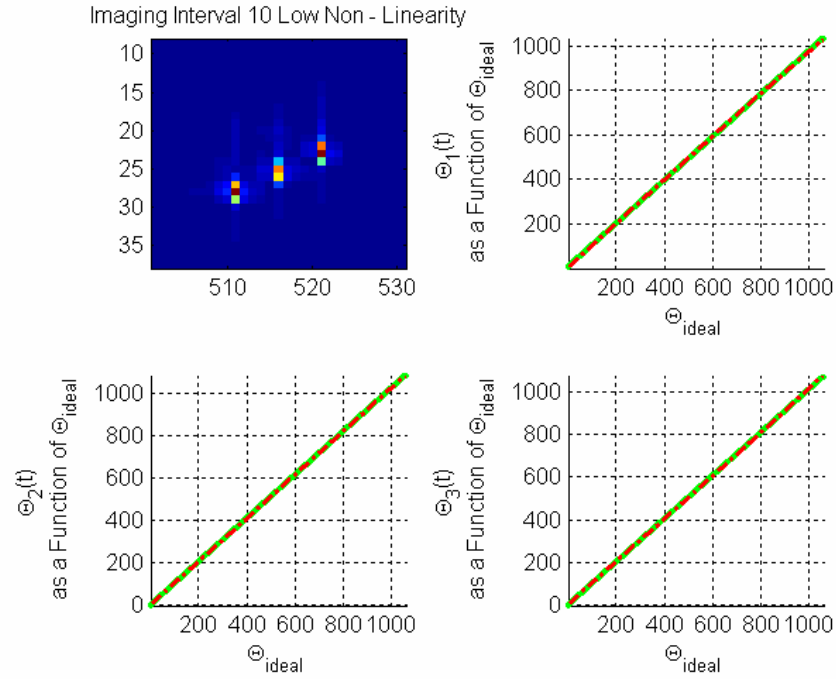


Figure 76. Imaging Interval 10 – Well Focused with Low Degree of Non-Linearity of Phases.

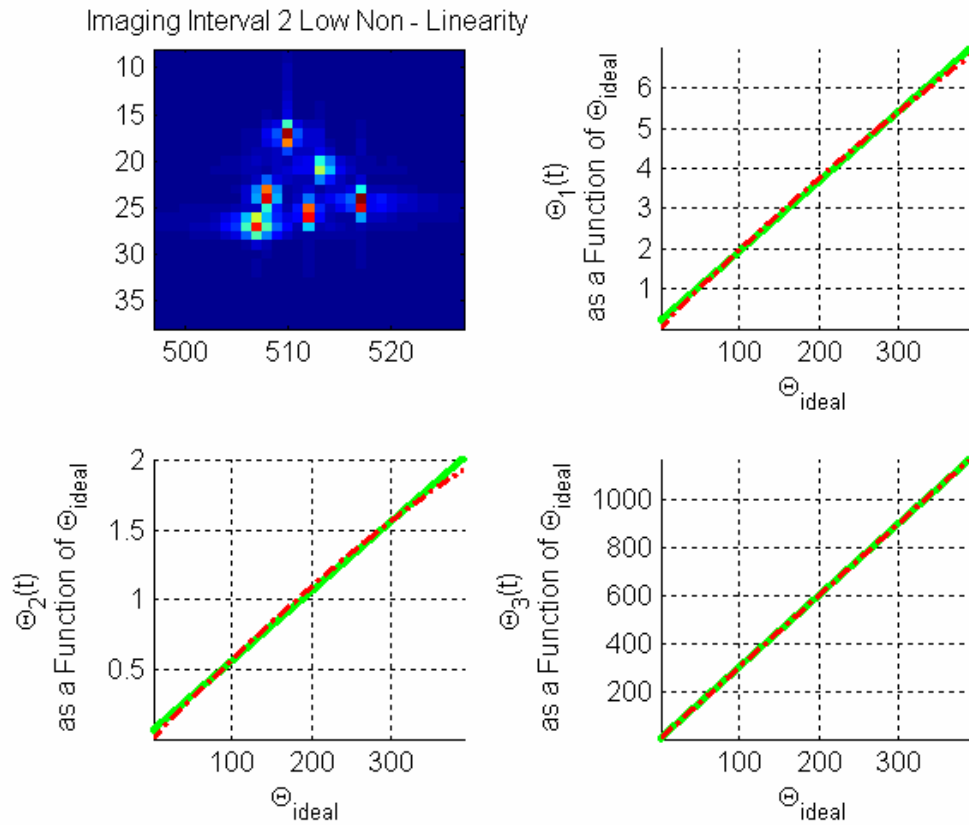


Figure 77. Imaging Interval 2 – Well Focused with Low Degree of Non-Linearity of Phases.

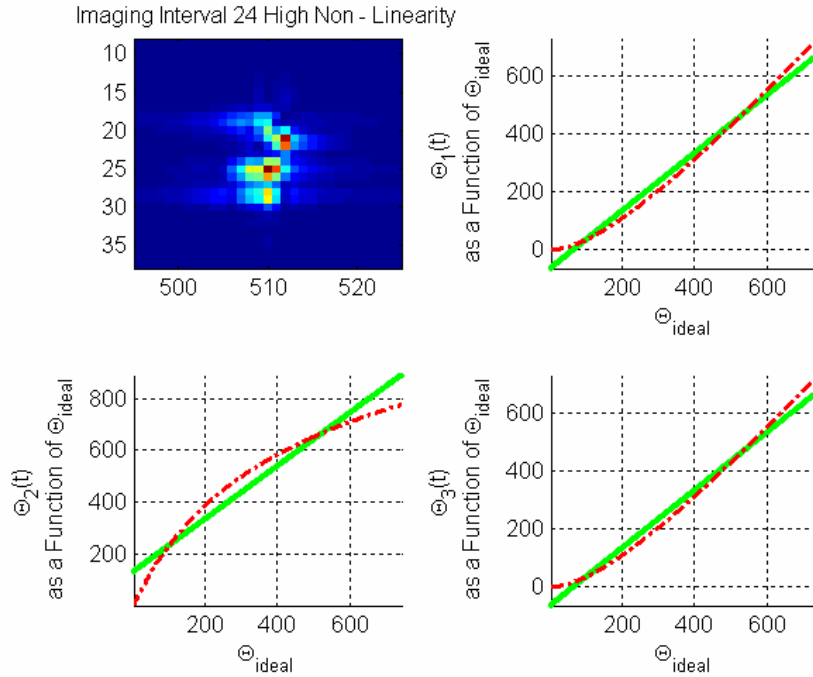


Figure 78. Imaging Interval 24 – Poorly Focused with High Degree of Non-Linearity of Phases.

5. Comparison between the Different Imaging Interval Sizes and the Degree of Motion Compensation

Now that the degree of linearity of phases has been generated for the two different imaging interval sizes and 2nd and 4th-degree motion compensation, it is possible to compare the results and infer which combination is better suited for rapid ISAR image focusing. The most straight-forward way to go about this is in the form of a table that lists the processing time required to produce each graph. Note that the total processing time is the time that it takes to bring every imaging interval in the 60,000 pulse 6 – Point Delta Wing experimental data set from their non-focused state to the point where their degree of 3D motion has been measured and a decision can be made with respect to whether or not the imaging interval should be focused or discarded. The results are shown in Table 2:

# of Pulses in Cross-range	Number of Images	Degree	Total Processing Time (seconds)	Processing Time per Image (seconds)
2048	29	2	561.8	19.3
2048	29	4	1710.1	58.9
1024	58	2	547.8	9.4
1024	58	4	1664.1	28.7

Table 2. Processing Time – Degree of Non-Linearity of Phases for Different Imaging Intervals.

The first thing that is noticeable from Table 2 is that, unsurprisingly, the processing time for the entire data set increases by over a factor of 3 when 4th-degree motion compensation is used instead of 2nd-degree motion compensation. Since there is only a marginal increase in the quality of the images (only imaging interval 13 with 2048 pulses is significantly affected), using higher order motion compensation for the 6 – Point Delta Wing experimental data set is a poor use of processing time.

This now leaves the 2048 pulse imaging interval with 2nd-degree motion compensation to be compared its 1024 pulse counterpart. Using the data set divided into 1024 pulse imaging intervals has two advantages over the 2048 pulse imaging intervals. First, it has twice as many images to work with so, when the AJTF PSO does not converge to a focused image (which will occasionally be the case when the PSO is set at an optimal combination of probability of convergence and run time), discarding an imaging interval is no great loss. Secondly, half of the processing time per image is helpful in getting images to the ISAR operator as quickly as possible. The data set with 2048 pulses in the cross-range has the advantage of producing better images with point scatterers that are well separated in Doppler. In an operational situation, these trade-offs will have to be balanced against each other. However, for the purposes of this thesis it was easier to set up 3D motion detection for the 2048 imaging intervals since 3D motion detection and the AJTF algorithm are very sensitive not only to range bin selection but as well to point scatterer selection.

C. CHAPTER SUMMARY

This Chapter presented Thayaparan's method of 3D motion detection after it had been adapted to the AJTF PSO. It was very effective and efficient at sorting between the good imaging interval which focused well and the poor imaging intervals that possessed 3D motion and could not be focused. The poor imaging intervals can simply be discarded and not presented to the ISAR operator. In addition, it was shown that for the 6 – Point Delta Wing experimental data set, 2nd-degree motion compensation is the most efficient used of computational time.

VI. CONCLUSION

A. SUMMARY OF RESULTS

1. AJTF GA and PSO Algorithms

The AJTF GA and PSO algorithms performed very well in their task of reducing the computational burden required by the AJTF algorithm when employing the exhaustive search for search parameter extraction. Typically, less than 50% of the cost function calculations were required for the GA and PSO algorithms to extract the translational and rotational motion compensation parameters which were able to focus the ISAR images in the cross-range. In addition, a new method of determining search parameter suitability, the automatic recognition of focus using the fast Fourier transform, was designed and proved very fast and extremely effective in providing accurate cost function returns that gave an accurate figure of merit on the suitability of solution space search parameters. When comparing the two different algorithms, the PSO algorithm consistently outperformed the GA and was able to converge to a focused image with less calculations of the cost function. Also of important note is that the use of the PSO algorithm in ISAR image focusing is a unique application of this evolutionary search technique which has not been published in the scientific literature.

2. 3D Motion Detection

Thayaparan's 3D motion detection algorithm in [12], after adaptation to this thesis' AJTF PSO algorithm and uniquely changing the measure of non-linearity of phases to the percentage of deviation from the line of least-squares fit, proved effective in separating good imaging intervals, possessing only 2D motion, from the poor imaging intervals that possessed a significant amount of 3D motion. Application of the AJTF PSO to the 3D detection algorithm significantly reduced computational time, especially since it was required that 3 phase functions be extracted from 3 different point scatterers. The AJTF PSO algorithm is a unique approach to the 3D motion detection problem and creates an algorithm which is both fast and accurate. It was also realized that the application of motion compensation greater than the 2nd-degree just adds to the computational burden associated with 3D motion detection without adding any significant improvement to 3D motion detection or image quality.

B. FOLLOW-ON WORK

For those interested in ISAR image formation, there are several areas of research that could be studied. First, an important, but classified, thesis topic could involve the AJTF PSO algorithm being tested against ISAR returns of actual military targets to verify that the motion compensation algorithm can be deployed against such targets of interest. Secondly, jet engine modulation (JEM) introduces severe time-variance into an ISAR return. Worthwhile research would look at ways of removing the image smearing due to JEM or classifying targets based on their JEM signature. A good reference for this research topic is [15].

APPENDIX MATLAB CODE

Listed in this appendix is the Matlab code used in this thesis. The order of listing is the GA code, followed by the PSO code and the 3D motion detection code. The code requires MatLab 6.5 or later and the signal processing toolbox to run.

A. GA CODE

```
function [fD1, fD2, FS, FST] = GA_Image_Focus(G, R1, R2, m_order1, m_order2, num_pop, num_gen, output)
% Written by Capt Wade Brinkman, Canadian Forces
% Oct 9, 2004
% Modified: 10 Nov 2004
% As partial fulfillment of the MSEE Degree at NPS
% Original DRDC version written by: Jonathan Waisman
% This version used the exhaustive search technique as described in
% thesis - see reference page for DRDC paper

% This algorithm will focus ISAR images using a Genetic Algorithm
% application of the AJTF algorithm.

% Input:
% G - signal (2D MxN array)
% R1 - a range bin that will be used for translational motion comp ( 1 to M )
% R2 - a range bin that will be used for rotational motion comp (1 to M);
% m_order1 - order of motion in R1
% m_order2 - order of motion in R2.
% num_pop - the number in the GA population
% num_gen - number of generations in the run
% output - 1 - output TF Transforms
%        0 - supresses the TF Transforms
% Output:
% fD1 - the doppler coefficients for the trans motion comp
% fD2 - the doppler coefficients for the rotational motion comp
% FS - final focused signal
% FST - signal after translational motion comp

format short;
[M , N] = size(G);    %M -range bins N - pulses

fD1 = abs(fft(G(R1,:)));
fD1 = find(fD1 == max(fD1));

fD2 = abs(fft(G(R2,:)));
fD2 = find(fD2 == max(fD2));

%find the factorials for 1..m_order1 & two
for k = 1:m_order1;
    Order1_fac(k) = factorial(k);
end

for k = 1:m_order2;
    Order2_fac(k) = factorial(k);
end

%look at the uncompensated signal;
if max(size(G)) == 256;
    srddi(G,R1,R2,1,256,1,64,1); colormap('hot');
else
```

```

    srdi(G,R1,R2,round(N/2)-20,round(N/2)+20,17,31,2);
end
%now we can look at the TF of the two range bins
if output ==1;
    figure;
    tfr_x1 = tfrsp(G(R1,:)); %stft
    tfr_x1 = [tfr_x1(N/2+1:N,:); tfr_x1(1:N/2,:)];
    imagesc(abs(tfr_x1));
    title(['Uncompensated - Range Bin # ',num2str(R1)]);
    colormap('hot');

    figure;
    tfr_x2 = tfrsp(G(R2,:)); %stft
    tfr_x2 = [tfr_x2(N/2+1:N,:); tfr_x2(1:N/2,:)];
    imagesc(abs(tfr_x2));
    title(['Uncompensated - Range Bin # ',num2str(R2)]);
    colormap('hot');
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%'Step 1: Translational motion compensation using the first range bin'

[fD1 FST] = GA_AJTF(G,R1,m_order1,num_pop,num_gen,0,Order1_fac,fD1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%'Step 2 - Display Translational Motion Compensation Results'
if max(size(G)) == 256;
    srdi(FST,R1,R2,1,256,1,64,1); colormap('hot');
else
    srdi(FST,R1,R2,round(N/2)-20,round(N/2)+20,17,31,1);
end

%display the 2 range bins after translational rotation

if output == 1
    X1 = FST(R1,:); %1st range bin
    tfr_x1 = tfrsp(X1); %stft
    tfr_x1 = [tfr_x1(N/2+1:N,:);tfr_x1(1:N/2,:)];
    figure; imagesc(abs(tfr_x1));
    title(['Range Bin ',num2str(R1),' after Translational Motion Compensation']);
    colormap('hot');

    X2 = FST(R2,:); %1st range bin
    tfr_x2 = tfrsp(X2); %stft
    tfr_x2 = [tfr_x2(N/2+1:N,:);tfr_x2(1:N/2,:)];
    figure; imagesc(abs(tfr_x2));
    title(['Range Bin ',num2str(R2),' after Translational Motion Compensation']);
    colormap('hot');
end;

%'Step 3 - Rotational motion compensation using the second range bin'

[fD2 FS] = GA_AJTF(FST,R2,m_order2,num_pop,num_gen,1,Order2_fac,fD2);

%'Step 4 - Display Rotational Motion Compensation Results'

if output == 1;
    X1 = FS(R1,:); %1st range bin
    tfr_x1 = tfrsp(X1); %stft
    tfr_x1 = [tfr_x1(N/2+1:N,:);tfr_x1(1:N/2,:)];
    figure; imagesc(abs(tfr_x1));
    title(['Range Bin ',num2str(R1),' after Rotational Motion Compensation']);

```

```

colormap('hot');

X2 = FS(R2,:); %1st range bin
tfr_x2 = tfrsp(X2'); %stft
tfr_x2=[tfr_x2(N/2+1:N,:);tfr_x2(1:N/2,:)];
figure; imagesc(abs(tfr_x2));
title(['Range Bin ',num2str(R2),' after Rotational Motion Compensation']);
colormap('hot');
end;

%finally the compensated image:
if max(size(G)) == 256;
    srdi(FS,R1,R2,1,256,1,64,1); colormap('hot');
else
    srdi(FS,R1,R2,round(N/2)-20,round(N/2)+20,17,31,2);
end

function [fD,NS] = GA_AJTF(SIG,R,m_order,num_pop,num_gen,ToR,Order_fac,fD);

%Capt Brinkman, Canadian Forces
%Oct 23, 2004
%Modified: 20 Nov 04
%this function is main Genetic Algorithm function that calls all the
%other GA functions
%The goal of this real valued GA algorithm is to converge very quickly to
%the doppler parameters of the basis function

%SIG: the signal to be analyzed;
%R: range bin
%m_order: order of motion
%num_pop: this is the population of our genetic pool
%num_gen: the number of max # of generations to run
%ToR: 0 - compensating translational motion
%      1 - compensating rotational motion
%Order_fac - precalculated factorials needed for the basis function
%fD - fI search parameter

%Output - fD - returns the doppler coefficients for the motion comp
%      NS - motion compensated signal

[rbins pulses] = size(SIG);

index = 1;

%Step 1: Find the initial Population and rank it
population = I_POP(m_order, num_pop,pulses,fD);
    %creation our initial population of size num_pop
    %with values from -pulses:pulses
[gen_fit] = FITNESS_FFT(population,SIG(R,:),ToR,Order_fac);
%find the fitness of the first population
mut_gen = num_gen;
while num_gen>0

    gen_score(index) = sum(gen_fit);
    %ranks the overall fitness of the population
    generation_value(index) = sum(gen_fit)/num_pop;
    %sum of the parents raw score - this should increase over time
    best_value(index) = max(gen_fit);
    %this is the first best value found;

%Step 2: Based on Suitability, Generate Children

```

```

children = CHILDS(population,gen_fit,gen_score(index));

%Step 3: Mutation
mut_prob = (0.3); %30% mutation rate.
children = MUTATION(children,mu_prob,pulses,m_order);

%Step 4: Determine Suitability of Children:
[fit_child] = FITNESS_FFT(children,SIG(R,:),ToR,Order_fac);

%Step 5: Reinsertion
%inserts the best of old population and new population into the new
%population
[population,gen_fit] = REINSERTION(population,children,gen_fit,fit_child);

num_gen = num_gen -1;
index = index +1;
end;

%pick out the best function;
best = population(find(gen_fit == max(gen_fit)),:);
fD = best(1,:);

%Now the new signal can be calculated
if ToR == 0; %translational motion
hp = basis2(fD,Order_fac,m_order,1,pulses,ToR);
if pulses<2%turn off for now
tfr_hp = abs(stft(hp));
tfr_hp = [tfr_hp(pulses/2+1:pulses,:); tfr_hp(1:pulses/2,:)];
figure;imagesc(abs(tfr_hp));
end;
NS = SIG.*conj(repmat(hp,rbins,1));
end;

if ToR == 1; %rotational motion comp
clear theta;
[dummy,theta]=basis2(fD,Order_fac,m_order,1,pulses,ToR);
spacing = ((max(theta)-min(theta))/pulses)*0.999;
uniform_samples = min(theta)+(1:pulses)*spacing;
NS = interp1q(theta',SIG',uniform_samples,:); %reformat the radar data
%so that is uniformly samples in theta
end;

figure;
plot(generation_value);
hold on; plot(best_value,'g.-');

function [population] = I_POP(m_order,num_pop,N,fD);

%function I_POP will create the initial population of possible solutions
%m_order - order of motion
%num_pop - max number in the population
%N - the values of the pop will be between -N:N;
%fD - is our f1 coefficient as determined by the FFT

%if m_order == 2;
% population = linspace(-N,N,num_pop)';
%elseif m_order>2;
% population = [linspace(-N,N,num_pop)' (-N + 2*N*rand(num_pop,m_order-2))];
%end;

population = -N + 2*N*rand(num_pop,m_order-1);
%guess at f2, f3, f4.....m_order as random vars

```

```

fD_p(1:num_pop,:) = fD;
population = [fD_p population];
    %add f1 to the population

function [gen_fit] = FITNESS_FFT(pop, X, ToR,fac);
format long
%Capt Brinkman, 16 Oct 2004
%Modified 20 Jan 2005
%this function generates the suitability of each
%parent in the current generation

%pop - matrix of the current population
%    - f1, f2, f3 etc depending on the degree;
%X - the range bin under examination
%ToR - 0 - translational
%    - 1 - rotational
%fac - precalculated factorials

%what we are doing here is trying to drive the slope
%of the scatter in the T - F plane to zero. When this
%is achieved, translational or rotational motion can be
%removed.

[num_pop, degree] = size(pop);

[a pulses] = size(X); %number of pulses in the range bin

if ToR == 0; %compensating translational motion:
    hp_vec = basis2(pop,fac,degree,num_pop,pulses,ToR);
        %hp_vec is the basis func for the pop
    X = repmat(X,num_pop,1);
    test_result = X .* conj(hp_vec);
    test = (abs(fft(test_result)).^2);%find the power spectral density of pop
    for k = 1:num_pop;
        test_fft = test(k,:);

        max_test = max(test_fft);
        avg_PSD = max_test/sum(test_fft); %Percentage of Power under max point

        gen_fit1(k,1)=avg_PSD; %max_sum;
        %when the percentage of energy in the dominant scatterer of the
        %range bin is a maximum, we have our best focus
    end;

    gen_fit = 100.^(gen_fit1);
    gen_value = gen_fit1;

    theta = [];
end;

if ToR ==1; %compensting rotational motion
    [dummy theta] = basis2(pop,fac,length(fac),num_pop,pulses,ToR);
    for k = 1:num_pop;
        spacing = ((max(theta(k,:))-min(theta(k,:)))/pulses)*0.999;
        uniform_samples = min(theta(k,:))+(1:pulses)*spacing;
        test_sig = interp1q(theta(k,:),X',uniform_samples)';
            %reformat the signal so that it is linear sample w/
            %theta not time.
        test_fft = abs(fft(test_sig)).^2; %find the power spectral density of the scatterer rb
        max_test = max(test_fft);
        avg_PSD = max_test/sum(test_fft); %Percentage of Power under max point
        gen_fit1(k,1)=avg_PSD; %max_sum;
        %when the percentage of energy in the dominant scatterer of the
        %range bin is a maximum, we have our best focus
    end;
end;

```

```

    end; %end for loop
end;

gen_fit = 100.^(gen_fit1);
    %the exponential provides separation between good values
    %and bad values. The higher the exponential, the greater the
    %GA searches around first global max found. The smaller, the
    %longer it takes the pop to base towards global max and a
    %global search is executed.
gen_value = gen_fit1;

function [children] = CHILDS(population,gen_fit,gen_score);
%Capt Wade Brinkman
%Canadian Forces - 20 Oct 04

%this function will select the best parents based on their assessed fitness
%and breed them in order to create the children;

%a roulette wheel selection method is used. This is discussed in detail in
%Goldberg and the GA toolbox manual

%population - the current GA pop that we are working with
%gen_fit - score between 1 and zero that assesses their fitness
%gen_score - the score for the current gen (sum(gen_fit));
[N degree]= size(population);
%Step 1: Assign Percentage of the roulette wheel

raw_score = floor((gen_fit./gen_score).*10000)/10000;
    %converts gen_fit to a percentage truncated at 4 decimals;
wheel = cumsum(raw_score);
    %the values progressively increase and this becomes our random
    %roulette wheel

%Step 2: find the parents chosen by the roulette wheel
select = max(wheel)*rand(1,length(gen_fit));
    %these are our pointers for the entire selection process
select = repmat(select,N,1);
    %convert to matrix
wheel = repmat(wheel,1,length(gen_fit));
    %convert to matrix for vector

choice = wheel>=select; %places a 1 where this is true
    %the index of the first value in each column corresponds to
    %a parent that has been chosen by the random roulette wheel

for index = 1:N;
    check = find(choice(:,index)==1);
    Pool(index,:)=population(check(1),:); %necessary for version 6
    %check = find(choice(:,index)==1,1,'first'); -works for version 7
    %Pool(index,:)=population(find(choice(:,index)==1,1,'first'),:);
    %this is our list of good parents that will produce good offspring
end;

pool1 = Pool(1:N/2,:);
pool2 = Pool(N/2+1:N,:); %break the pool up into two equal matrix

%the first one in the population, after first gen best parent, is
%guaranteed two spots - one in each pool

```

```

pool1(1,:) = population(1,:);
pool2(1,:) = population(1,:);

index1 = randperm(N/2);
index2= randperm(N/2);
    %generate a random order;

alpha=[ones(N/2,1) (0 + 1.*rand(N/2,degree-1))]; %alpha is our scaling factor
%alpha = repmat(alpha,1,degree); %reformat it into a N/2 - degree
    %array to allow for vectorization
child1 = alpha.*pool1(index1,:)+(1-alpha).*pool2(index2,:);
child2 = (1-alpha).*pool1(index1,:)+(alpha).*pool2(index2,:);
%calculate our children using the method described in
%thesis and Junfei Li and Hao Ling's paper - see references

children = [child1;child2];
    %combine our children array and return.

function [population,gen_fit,gen_value] = REINSERTION(population,children,gen_fit,...
    fit_child,gen_value,value_child);
%Capt Wade Brinkman
%Modified: 20 Nov 04
%Input
%population - old population
%children - the children of the current population
%gen_fit - fitness of each element in population
%fit_child - fitness of the children
%Output
%population - new population
%gen_fit - new generation fitness
%gen_value - new generation value
%This file will create a new population based on fitness

%first step is to combine the two vectors
temp_pop = [population;children];
temp_fit = [gen_fit;fit_child];

for k=1:length(population)/2;
    max_fit = max(temp_fit); %find the maximum fitness
    max_index = find(temp_fit == max_fit); %maximum index
    max_index = max_index(1); %ensures one value is found

    population(k,:) = temp_pop(max_index,:);
    gen_fit(k,:) = temp_fit(max_index);

    temp_fit(max_index(1)) = 0;
end

index_new = find(temp_fit~=0);
new_fit(1:length(index_new))=temp_fit(index_new);
new_pop(1:length(index_new),:)= temp_pop(index_new,:);

for k = (length(population)/2+1):length(population);
    index_rand = round(1+(length(population)-1)*rand);
    population(k,:) = new_pop(index_rand,:);
    gen_fit(k,:) = new_fit(index_rand);
end;

%this is our new population with our new measures of fitness;

function [show_image] = srdi(sig,rb1,rb2,xmin,xmax,ymin,ymax,itype);
%si = srdi(FS,28,19,505,520,17,31,2); will work for fsmall data set
rbin1 = sig(rb1,:);
rbin2 = sig(rb2,:);

```



```

show_image = rdi(sig);

subplot(2,2,[1 3]);
imagesc(show_image);
ylabel('Down Range - Range Bins');
xlabel('Cross Range - Doppler Bins');

axis([xmin xmax ymin ymax]);
if itype == 0;
    title('Uncompensated Image');
elseif itype == 1;
    title('Image After Translational Motion Compensation');
elseif itype == 2;
    title('Final Image after Rotational Motion Compensation');
end

f_rbin1 = fliplr(abs(fftshift(fft(rbin1))).^2);
f_rbin2 = fliplr(abs(fftshift(fft(rbin2))).^2);
n = 2:(length(rbin1)+1);

subplot(2,2,2);
plot(n,f_rbin1,'r-');
title(['Power Spectrum Range Bin ',num2str(rb1)]);
axis([xmin xmax min(f_rbin1) max(f_rbin1)]);
ylabel('Amplitude');
xlabel('Cross Range - Doppler Bins');

subplot(2,2,4);
plot(n,f_rbin2,'r-');
title(['Power Spectrum Range Bin ',num2str(rb2)]);
axis([xmin xmax min(f_rbin2) max(f_rbin2)]);
ylabel('Amplitude');
xlabel('Cross Range - Doppler Bins');

function [hp,theta] = basis2(pop,fac,degree,num_pop,pulses,ToR)
%Capt Wade Brinkman
%16 Oct 04
%Modified 20 Jan 05
%this function will calculate the basis function for our current population
%the calculation will be done in parallel to speed up comp time
%this version removes the for loops which cuts down the run time by half
%this is important since we are trying to drive the run-time of the GA and Swarm to
%almost zero
%Input:
%pop - current population
%fac - the factorials used in the calculation
%degree - the degree of motion
%num_pop - the number in current population
%pulses - # of pulse in cross range
%ToR - Translational or rotational 0 - theta = []; 1 - hp = [];
%Output:
%hp - the basis function of each mbr of pop
%theta - the phases associated with each mbr of pop
hp = zeros(num_pop, pulses);

t = ((1:pulses)./pulses)'; %t = [(1 2 3 4 ... pulses)./pulses]' - column vector
t_temp = t;
for i = 2:degree
    t = [t (t_temp).^i];
    %t^1 forms a column of t, t^2 forms the next...used to vectorize
end

pop_fac = pop./(repmat(fac,num_pop,1)); %predividing the population by its factorials
theta = (pop_fac(1:num_pop,:) * t(1:pulses,:));

```

```

if ToR == 1;
    hp = [];
else
    hp(1:num_pop,1:pulses) = exp(theta.*(-j*2*pi));
    theta = [];
end;
%finally, this calculates the basis function for the entire population.
function f = rdi(s)
% generates a range doppler image of the signal s

f = fft(s');
[n,m] = size(f);
f = [f(round(n/2)+1:n,:); f(1:round(n/2),:)]; % equivalent to fftshift

f = abs(f/1);
f = (f./max(max(f)));

figure; imagesc(abs(f));
colormap('jet');

```

B. PSO CODE

```

function [FS fDT fDR] = Swarm_Image_Focus_3D(G, R1,D_T, RFocus,D_R,num_particles,cycles)
% Written by Capt Wade Brinkman, Canadian Forces
% Nov 13, 2004
% Modified: 12 Mar 2005
% As partial fulfillment of the MSEE Degree at NPS
% Original DRDC version written by: Jonathan Waisman
% The DRDC version used the exhaustive search technique as described in
% thesis - see reference page for DRDC paper
% This algorithm will focus ISAR images using a Swarm Algorithm
% application of the AJTF algorithm.
% This algorithm uses a Swarm Algorithm as described in Jacob Robinson
% and Daniel Boeringer papers - see reference page - with modifications
% to fit the AJTF algorithm
% This algorithm uses several scatterers from the same range bin for
% rotational motion compensation.
% Input:
% G - signal (2D MxN array)
% R1 - a range bin that will be used for translational motion comp
% D_T - degree of translational motion compensation
% RFocus - [Range Bin, # Scatterers, Range Bin, # Scatterers,...]
% list of range bin and number of scatterers for rotational motion
% compensation
% D_R - degree of rotation motion compensation
% num_particles - the number of particles in the Swarm - identical to
% population
% cycles - number of times the Swarm's position are updated - identical
% to number of generations in GA

% Output:
% fDT - the doppler coefficients for the translational motion
% compensation
% fDR - the doppler coefficients for the rotational motion compensation
% FS - final focused signal

format short;
[M , N] = size(G); %M -range bins N - pulses
N_Rot = length(RFocus);
if mod(N_Rot,2)~=0;
    disp('Rotational Range Bin Matrix incorrectly formatted!');
    FS = G;fDT = []; fDR = [];
    return
end

```

```

N_RSB=N_Rot/2; %number of range bins for rot mot comp
RFocus = reshape(RFocus,2,N_RSB);

for k = 1:D_T;%find the factorials
    Order1_fac(k) = factorial(k);
end

for k = 1:D_R;
    Order2_fac(k) = factorial(k);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%'Step 1: Translational motion compensation using the first range bin'
%'For this step, the Swarm_AJTF_3D will return the Doppler Frequencies, but not
%'focus the image

[fDT] = Swarm_AJTF_3D(G,R1,D_T,num_particles,cycles,0,Order1_fac);
FST = tr_focus_3D(G,fDT,0);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FST_t = FST;
%'Step 2 - Rotational motion compensation using the Range Bins
%'and number of scatterers found in RFocus
r_fDR = 1; %index for the rotational doppler
for k = 1:N_RSB
    for g = 1:RFocus(2,k);
        [fDR(r_fDR,:)] = Swarm_AJTF_3D(FST_t,RFocus(1,k),D_R,num_particles,cycles,1,Order2_fac);
        if RFocus(2,k)>1 && g<RFocus(2,k)
            X_t = fft(FST_t(RFocus(1,k),:));
            index = fDR(r_fDR,1)+(-3:3);
            for kk = 1:length(index);
                if index(kk)<1 X_t(N+index(kk))=0;
                elseif index(kk)>N X_t(index(kk)-N)=0; %remove used scatterer from Range Bin
                else X_t(index(kk))=0; end;
            end;
            FST_t(RFocus(1,k),:) = ifft(X_t);
        end;
        r_fDR = r_fDR + 1;
    end
end;

FD = mean(fDR,1);
FS = tr_focus_3D(FST,FD,1);
function [fD] = Swarm_AJTF_3D(SIG,R,m_order,num_particles,cycles,ToR,Order_fac,fD);

%Capt Brinkman, Canadian Forces
%Nov 13, 2004
%Modified 20 Jan 05
%this function is main Swarm Algorithm function that calls all the
%other Swarm functions. It is called once for translational and once for
%rotational motion compensation.
%The goal of this real valued Swarm algorithm is to coverge very quickly to
%the doppler parameters of the basis function
%this is a line search - it will find f1, fix f1, find f2, fix f2 find f3
%etc. Necessary since solution space grows by N^m_order
%SIG: the signal to be analysed;
%R: range bin
%m_order: order of motion
%num_particles: this is the number of partilces in our Swarm
%cycles: max # of generations to run
%ToR: 0 - compensating translational motion
%      1 - compensating rotational motion
%Order_fac - precalculated factorials needed for the basis function
%fD = f1
%Output - fD - returns the doppler coefficients for the motion comp

```

```

fD = abs(fft(SIG(R,:))); fD = find(fD == max(fD));
[rbins pulses] = size(SIG);
num_cycles = cycles;
GFit_Old = 0;

current_f = 2; %current parameter being searched
Swarm=zeros(num_particles,1);
Swarm(1:num_particles,1)=fD];

while current_f<=m_order

%Step 1: Find the initial Swarm and evaluate fitness
[Swarm,velocity] = l_Swarm(Swarm,m_order, num_particles,pulses,current_f);
    %creation our initial swarm and velocity
    %with values from -pulses:pulses and intial velocities from
    %-pulses:pulses
%eval the fitness
[part_fit] = FITNESS_FFT(Swarm,SIG(R,:),ToR,Order_fac(1:current_f),current_f);
LBest = Swarm; %these are the local 'bests' found by each particle. For the first cycle, by default it is
    %equivalent to the first Swarm
LFit = part_fit; %LFit is the fitness of the local best
GFit = max(LFit); %this is the maximum global fitness that was found
GBest = LBest(find(LFit==max(LFit)),:); %this is current global best.....
GFit = GFit(1); GBest = GBest(1,:);
index = 1;
part_score(index) = sum(LFit)./num_particles;
    %ranks the overall fitness of the population
best_value(index) = GFit;
    %this is the current global maximum;

best_count = cycles;
o_Best = GFit; %these two variables are used to cause the algorithm to exit if no improvement has been made after x - cycles
%figure;
while cycles>0
    %hold on; plot(index,LBest(:,2),'g+');hold on;plot(index,GBest(1,2),'b>');hold on; plot(index,Swarm(:,2),'c. ');
    cycles = cycles -1;
    index = index +1;

%Step 2: Update velocities:
[velocity]=v_update(velocity,LBest,GBest,Swarm,num_particles,current_f,pulses);

%Step 3: Update positions
[Swarm,velocity]=p_update(Swarm,velocity,num_particles,pulses);

%Step 4: determine fitness of the Swarm
[part_fit] = FITNESS_FFT(Swarm,SIG(R,:),ToR,Order_fac(1:current_f),current_f);

%Step 5: determine LBest and GBest
[LBest, LFit, GBest, GFit] = swap_best(LBest, LFit, GBest, GFit, Swarm,part_fit,current_f);

part_score(index) = sum(LFit)./num_particles;
    %ranks the overall fitness of the population
best_value(index) = GFit;
    %this is the current global maximum;

end;

if GFit_Old >= GFit
    GBest(current_f)=0; %if the Fitness of this degree is less than the previous, set this parameter
    %equal to zero.
else

```

```

    GFit_Old = GFit;
end

current_f = current_f+1;
cycles = num_cycles;
Swarm= repmat(GBest,num_particles,1);

end;

fD = GBest;

function [Swarm, velocity] = I_Swarm(Swarm,m_order,num_particles,N,current_f);
%Capt Wade Brinkman, Canadian Forces
%14 Nov 04
%Modified 12 Jan 05
%function I_Swarm will ceate the initial particles of possible solutions
%and assign velocities random in both mag and direction to each particle

%Swarm - current Swarm - will be expanded
%m_order - order of motion
%num_particles- max number in the particles in the Swarm
%N - the values of the pop will be between -N:N;
%fD - is our f1 coefficient as determined by the FFT

Swarm =[Swarm (-N + 2*N*rand(num_particles,1))];
%guess at f2, f3, f4.....m_order as random vars
velocity = (2*N)*rand(num_particles,1);
%magnitude of the velocites
v_dir = rand(num_particles,1)-0.5;
%direction of the particle +ve or -ve
index1 = find(v_dir>=0);
index2 = find(v_dir<0);
v_dir(index1)=1;
v_dir(index2)=-1;

velocity = velocity.*v_dir;
velocity = [zeros(num_particles,current_f-1) velocity];
%the f1 velocity will always be fixed at zero.

function [velocity] = v_update(velocity,LBest,GBest,Swarm,num_particles,m_order,pulses);
%written by Capt Wade Brinkman, Canadian Forces
%Created 14 Nov 2004
%This function will update the velocities based on the particles
%current velocity and the pull of the the Local Best and the GBest
%Input:
%velocity - current velocity vectors
%LBest - matrix of local best discoveries
%GBest - the best discovery
%Swarm - current search location of the particles
%num_particles - number of particles in the swarm
%m_order - order of motion
%pulses - number of pulses - also corresponds to max velocity
%Output:
%velocity - updated velocity vector

K(1:num_particles,1:m_order)=0.729; %constriction factor
rho1 = 2.05;
rho2 = 2.05;
r_vec1 = rand(num_particles,m_order);
r_vec2 = rand(num_particles,m_order);
Global_Best=repmat(GBest,num_particles,1);

```

```

velocity = K.*(velocity + 2.05.*r_vec1.*(LBest-Swarm)+2.05.*r_vec2.*(Global_Best-Swarm));
%this sets the velocity vector for the particles
%Now, check to make sure that no particle exceeded the max velocity;
[R C V] = find(velocity>2*pulses);
if length(V)>0;
    for k=1:length(R)
        velocity(R(k),C(k)) = 2*pulses;
    end
end

[R C V] = find(velocity<-2*pulses);
if length(R)>0
    for k=1:length(R)
        velocity(R(k),C(k)) = -2*pulses;
    end
end

function [Swarm,velocity]=p_update(Swarm,velocity,num_particles,pulses);
%Programmed by Capt Wade Brinkman, Canadian Forces
%14 Nov 2004
%based on the new velocities, the positions will be updated. Any particle
%outside the solution space of +/- pulses will have its velocity in that
%dimension zeroed
%Input:
%Swarm: The particle Swarm
%velocity: Particles Current velocity
%num_particles: Number of particles in the Swarm
%pulses: number of pulses in signal which also defines the bounds
%       of the solution space

Swarm = Swarm + velocity;
[R C V] = find(Swarm>pulses);

if length(V)>0; %+ve wall
    for k = 1:length(V);
        velocity(R(k),C(k))=-pulses*0.1*rand; %rebounding wall
        Swarm(R(k),C(k))=pulses;
    end
end

[R C V] = find(Swarm<-pulses);

if length(V)>0; %-ve wall
    for k = 1:length(V);
        velocity(R(k),C(k))=pulses*0.1*rand; %rebounding wall
        Swarm(R(k),C(k))=-pulses;
    end
end;

function [LBest, LFit, GBest, GFit] = swap_best(LBest, LFit, GBest, GFit, Swarm,part_fit,m_order);
%Written by Captain Wade Brinkman, Canadian Forces
%14 Nov 2004
%This function will update the local and global bests as required
%Inputs:
%LBest - matrix of local best parameters
%LFit - a column vector of the fitness of the local bests
%GBest - is the parameters of the global best
%GFit - fitness of the global best;
%Swarm - current parameters of the particle swarm
%part_fit - current Swarm's fitness
%m_order - order of motion
%Output: LBest, LFit, GBest, GFit - updated parameters

Max_Swarm_fit = max(part_fit); %the best of the current particles
Max_Swarm = Swarm(find(part_fit==Max_Swarm_fit,:));
if Max_Swarm_fit>GFit;

```

```

    GFit = Max_Swarm_fit(1);
    GBest = Max_Swarm(1,:);
end;

vec1 = LFit>=part_fit; %is current LBest better than current location
vec2 = part_fit>LFit; %is current location better than LBest

LFit = LFit.*vec1+part_fit.*vec2; %replaces the better current positions and LFit
LBest = LBest.*repmat(vec1,1,m_order)+Swarm.*repmat(vec2,1,m_order); %places the better position in Local Best

function [Focused_Signal] = tr_focus_3D(signal,fD,ToR);
%Capt Wade Brinkman
%15 Nov 2004
%this is a utility function tha will perform translational
%or rotational motion compensation on the signal

[rbins pulses] = size(signal);
for k = 1:length(fD);
    o_fac(k) = factorial(k);
end

[hp theta] = basis2(fD,o_fac,length(fD),1,pulses,ToR);

if ToR == 0
    Focused_Signal = signal.*conj(repmat(hp,rbins,1));
else
    spacing = (max(theta)-min(theta))/pulses*0.99999;
    uniform_samples = min(theta)+(1:pulses)*spacing;
    Focused_Signal = interp1q(theta.',signal.',uniform_samples.').';
end;

```

C. 3D MOTION DETECTION CODE

```

function [NL]=MDetect_3D(namedata,num_pulses,load_data,save_data,output,Num_S,ord1,ord2,cmovie);
%function [NL]=MDetect_3D('data',2^11,1,0,1,2,2,2,0);
%Capt Wade Brinkman
%Created 6 Feb 2005
%Modified 12 March 2005
%This is a program that will detect the presense of 3D motion
%It uses the EAJTF Algorithm {as described by Thayaparan's unpublished work}
%with the Swarm algorithm as its search engine
%Linearity of Phases is the Average Linearity of Phases Method
%{as described by Thayaparan's unpublished work}
%INPUT
% namedata - the name of the data set {without the imaging interval
%         number}
% num_pulses - number of ISAR pulses in the cross-range
%         - either 2^10 or 2^11
% load_data - if 1 - fD and theta already exist
% save_data - save the fD and theta
% output - output all images
% Num_S - number of Scatterers in first range bin {default 1 in second}
% ord1 - order of motion comp - Translational
% ord2 - order of motion comp - Rotational
% cmovie - create a movie
%OUTPUT
% NL - Degree of Non-Linearity

fflag = 1;
num_images = floor(60000/num_pulses);

%first set the directory of choices
if (num_pulses == 2^11)&&(ord1==2);
    cd LP2048_2D;
    directory = cd; cd ..; %sets our active directory where fD and theta are stored

```

```

    rb_file = 'rb_focus_3D'; %this is the file that contains the range bins used
    temp_load = cat(2,directory, '/',rb_file);
    load(temp_load); %the rb_focus_3D contains the range bins for 3D motion detection
    rb_focus = rb_focus_3D; clear rb_focus_3D;
elseif (num_pulses == 2^11)&&(ord1==4);
    cd LP2048_4D;
    directory = cd; cd ..; %sets our active directory where fD and theta are stored
    rb_file = 'rb_focus_3D'; %this is the file that contains the range bins used
    temp_load = cat(2,directory, '/',rb_file);
    load(temp_load); %the rb_focus_3D contains the range bins for 3D motion detection
    rb_focus = rb_focus_3D; clear rb_focus_3D;
elseif (num_pulses == 2^10)&&(ord1==2);
    cd LP1024_2D;
    directory = cd; cd ..; %sets our active directory where fD and theta are stored
    rb_file = 'rb_focus2'; %this is the file that contains the range bins used
    temp_load = cat(2,directory, '/',rb_file);
    load(temp_load); %the rb_focus_3D contains the range bins for 3D motion detection
    rb_focus = rb_focus2; clear rb_focus2;
elseif (num_pulses == 2^10)&&(ord1==4);
    cd LP1024_4D;
    directory = cd; cd ..; %sets our active directory where fD and theta are stored
    rb_file = 'rb_focus2'; %this is the file that contains the range bins used
    temp_load = cat(2,directory, '/',rb_file);
    load(temp_load); %the rb_focus_3D contains the range bins for 3D motion detection
    rb_focus = rb_focus2; clear rb_focus2;
else
    disp('Must choose 2^11 or 2^10 pulses or Order of Motion Error equal to 2 or 4');
    return
end;

for k = 1:num_images;
    if load_data == 0; %need to generate new fDs and phases
        if fflag == 1; fflag = 0; fac = zeros(1,ord2); for g = 1:ord2; fac(g) = factorial(g); end; end; %precalc factorials
        image_int = cat(2,directory, '/',namedata,int2str(k-1)); %create string containing name of imaging interval
        load(image_int); %load imaging interval into work space
        [FS fDT fDR] = Swarm_Image_Focus_3D(dat,rb_focus(k,1),ord1,[rb_focus(k,2),Num_S,rb_focus(k,3),1],...
            ord2,20,100);

        if output==1
            rdi(FS);
            title(['Focused Image Imaging Interval ',num2str(k-1),' - ',num2str(num_pulses),' Pulses in Cross Range']);
            xlabel('Cross Range - Doppler Bins');
            ylabel('Down Range - Range Bins');
        end
        [dum ord2] = size(fDR); f = [fDR]; [m n] = size(f);
        [hp,theta] = basis2(f,fac,ord2,m,num_pulses,1);
    elseif load_data == 1; %data already exists
        doppler = cat(2,directory, '/',fD,int2str(k-1));
        load(doppler);
        [r c] = size(fD);
        fDT = fD(1,:);
        fDR = fD(2:r,:);
        if output == 1;
            image_int = cat(2,directory, '/',namedata,int2str(k-1));
            load(image_int);
            FST = tr_focus_3D(dat,fDT,0);
            FS = tr_focus(FST,mean(fDR,1),1);
            title(['Focused Image Imaging Interval ',num2str(k-1),' - ',num2str(num_pulses),' Pulses in Cross Range']);
            xlabel('Cross Range - Doppler Bins');
            ylabel('Down Range - Range Bins');
        end;
        [dum ord2]=size(fDR); %detect number of scatterers and order
        phases = cat(2,directory, '/',theta,int2str(k-1));
        load(phases);
        if fflag == 1; fflag = 0; fac = zeros(1,ord2); for g = 1:ord2; fac(g) = factorial(g); end; end; %precalc factorial
        f = [fDR]; [m n] = size(f);
    end;
    [NL(k)] = E_DEG_OF_NONLINEARITY2( theta ,f,min(size(theta)),fac,0 );
    if save_data == 1;
        doppler = cat(2,directory, '/',fD,int2str(k-1));

```



```

    fD = [fDT;fDR];
    save(doppler,'fD');
    phases = cat(2,directory,'/', 'theta',int2str(k-1));
    save(phases,'theta');
end;
end;

NL_norm = NL;
figure; plot(0:(num_images-1),NL_norm,'r+-','linewidth',2); grid on; axis tight
cutoff_line(1:(num_images)) = mean(NL);
%hold on; plot(0:(num_images-1),cutoff_line,'b-','linewidth',2); hold off;
xlabel('Imaging Interval');
ylabel('Degree of Non-Linearity');
title({'Degree of Non-Linearity in Each Imaging Interval',...
    ['6 - Point Delta Wing Data Set with ',num2str(num_pulses),...
    ' Pulses and Degree of Motion Compensation - ',num2str(ord1)]});

%now we can display the good images
%and make a movie if the option is selected
[r c v] = find(NL_norm<=.010*mean(NL));
for k = 1:length(c);
    image_int = cat(2,directory,'/',namedata,int2str(c(k)-1));
    load(image_int);
    doppler = cat(2,directory,'/', 'fD',int2str(c(k)-1));
    load(doppler);
    [row col] = size(fD);
    fDT = fD(1,:);
    fDR = fD(2:row,:);
    fD = mean(fDR,1);
    FST = tr_focus_3D(dat,fDT,0);
    FS = tr_focus(FST,fD,1);
    image = rdi2(FS);
    bound = image>.3;
    [r2 c2 v2] = find(bound==1);
    center_r = round((max(r2)+min(r2))/2);
    center_c = round((max(c2)+min(c2))/2);
    axis([center_c-15 center_c+15 8 38]);
    if cmovie ~=1; title(['Imaging Interval: ',num2str(c(k)-1)]); end;
    if cmovie ==1;
        axis off;
        pause(2);
        g_o4_NL(k) = getframe(gcf);
    end
end

if cmovie == 1; save g_o4_NL g_o4_NL; end;

%now we are going to plot out the three best and three worst imaging
%intervals
NL_t = NL;
for k = 1:3 %find the three best
    temp = find(NL_t == min(NL_t));
    best_image(k) = temp-1; NL_t(temp) = NaN;
end

for k = 1:3 %find the three worst
    temp = find(NL_t == max(NL_t));
    worst_image(k) = temp-1; NL_t(temp) = NaN;
end

%now plot out the 3 best and 3 worst along with their non-linearity plots

for k = 1:3
    figure;
    image_int = cat(2,directory,'/',namedata,int2str(best_image(k)));
    load(image_int);

```

```

    phases = cat(2,directory,'/', 'theta',int2str(best_image(k)));
    load(phases);
    doppler = cat(2,directory,'/', 'fD',int2str(best_image(k)));
    load(doppler);
    [row col] = size(fD);
    fDT = fD(1,:);
    fDR = fD(2:row,:); fD = mean(fDR,1);
    FST = tr_focus_3D(dat,fDT,0);
    FS = tr_focus_3D(FST,fD,1);
    im_int = rdi2(FS);
    subplot(2,2,1);
    imagesc(im_int);
    title(['Imaging Interval ',num2str(best_image(k)), ' Low Non - Linearity']);
    bound = im_int>.3;
    [r2 c2 v2] = find(bound==1);
    center_r = round((max(r2)+min(r2))/2);
    center_c = round((max(c2)+min(c2))/2);
    axis([center_c-15 center_c+15 8 38]);
    [test] = E_DEG_OF_NONLINEARITY2( theta ,fDR,min(size(theta)),fac,1 );
end;

for k = 1:3
    figure;
    image_int = cat(2,directory,'/', namedata,int2str(worst_image(k)));
    load(image_int);
    phases = cat(2,directory,'/', 'theta',int2str(worst_image(k)));
    load(phases);
    doppler = cat(2,directory,'/', 'fD',int2str(worst_image(k)));
    load(doppler);
    [row col] = size(fD);
    fDT = fD(1,:);
    fDR = fD(2:row,:); fD = mean(fDR,1);
    FST = tr_focus_3D(dat,fDT,0);
    FS = tr_focus_3D(FST,fD,1);
    im_int = rdi2(FS);
    subplot(2,2,1);
    imagesc(im_int);
    title(['Imaging Interval ',num2str(worst_image(k)), ' High Non - Linearity']);
    bound = im_int>.3;
    [r2 c2 v2] = find(bound==1);
    center_r = round((max(r2)+min(r2))/2);
    center_c = round((max(c2)+min(c2))/2);
    axis([center_c-15 center_c+15 8 38]);
    [test] = E_DEG_OF_NONLINEARITY2( theta ,fDR,min(size(theta)),fac,1 );
end;

function [AVGN] = E_DEG_OF_NONLINEARITY2( P , fD , L ,fac,OP)
% AVGN = E_DEG_OF_NONLINEARITY2( P , fD , L ) -
% Finds the presence of 3D motion in each imaging interval. And returns
% the following.
%
% INPUT(S):
% P - the phase function of an imaging interval
% fD - doppler coefficients for this phase function (P can be generated
% from fD)
% L - number of point scatterers chosen for analysis
%
% OUTPUT(S):
% AVGN - degree of 3D motion for the imaging interval that defined the
% phase function P. Note that AVGN is the average degree of
% non-linearity (or in other words the degree of 3D motion) for the non-linearities
% for every pair of phase functions defined in P.
%
% Author : Jonathan Waisman
% Modified by Capt Wade Brinkman, Canadian Forces
% First Modified: 9 Feb 05
% Last Modified: 29 Mar 05

[H , N] = size( P );
[B , ORDER] = size( fD );

```

```

Nij = zeros( 1 , L );
AVGN = 0;
AProj = 0;

%ideal doppler coef will be average
i_fD = mean(fD,1);
%ideal phase function
[hp i_theta]=basis2(i_fD,fac,ORDER,1,N,1);
aa = max(max([i_theta;P]));
for i = 1:L
    % generate the non-linearity between the IdealP and P(i,:)
    [Nij( i )] = E_DEG_OF_NONLINEARITY1( i_theta , P(i,:),OP,i+1 );
    % sum the non-linearities for averaging
    AVGN = AVGN + Nij( i );
end

AVGN = AVGN / L;
function [ N12 Proj] = E_DEG_OF_NONLINEARITY1(i_P , r_P,OP,index )
% [ N12 ] = DEG_OF_NONLINEARITY1( i_P , r_P ) - calculates the degree of
% non-linearity between i_P and r_P using a least squares approach.
%
% INPUT(S):
% r_P,i_P - 2 phase functions
%
% OUTPUT(S):
% N12 - degree of nonlinearity between these 2 particular phase functions
%
% Author : Jonathan Waisman
% Modified by Capt Wade Brinkman
% Last Modified: 29 March 05
% i_P(t) = a *r_P(t) + n(t) is the relation of 2 point scatterers.

p=polyfit(i_P,r_P,1);
p = fliplr( p );

N = length( i_P );
k = length( p );
func = zeros( 1 , N);
t = linspace(1,max(i_P),N);
for x = 1:N
    for i = 1:k
        func( x ) = func( x ) + p( i ) * ( i_P(x) ^ ( i - 1 ) );
    end
end

% p = [p(1),p(2)] where least squares approximation func(t) = p(1) + p(2)*t
% but func is the best fit linear approximation to i_P, and thus:
% the degree of nonlinearity is the difference in the plot of
% plot(i_P,func) and plot(i_P,r_P);

N12 = 0;

P = r_P;
FUNC = func;
for x = 1:N
    N12 = N12 + abs(P(x)-FUNC(x))./abs(FUNC(x));
end

if OP == 1
    subplot(2,2,index); hold on;

```

```

plot(i_P,func,'g.-','linewidth',2);hold on;
plot(i_P,r_P,'r.-','linewidth',2);
xlabel('\Theta_i_d_e_a_l');
ylabel({'\Theta_',num2str(index-1),'(t)'],' as a Function of \Theta_i_d_e_a_l');
grid on; axis tight
end;

```

%to see the degree of non-linearity, enter this line into the command line:

```

%figure; plot(i_P,func,'g.-','linewidth',2);hold on;
%plot(i_P,r_P,'r.-','linewidth',2);

```

```

% The degree of 3D motion as the deviation from a linear
% relationship between theta and phi over the dwell intervall
% is directly related to N by a constant Beta. Thus the degree of
% nonlinearity will give a degree of 3D motion!

```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

- [1] Thayaparan, T., S. K. Wong and E. Riseborough, "Focusing ISAR Images Using Adaptive Joint Time-Frequency Algorithm on Simulated and Experimental Radar Data," Defence R&D Canada – Ottawa, March 2003. Available at <http://www.drdc-rddc.dnd.ca/> (Date Accessed 5 December 2003)
- [2] Wang, Y., H. Ling and V. C. Chen, "ISAR Motion Compensation via Adaptive Joint Time-Frequency Technique," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 2, pp. 670 – 677, April 1998.
- [3] Chen, V. C. and H. Ling, *Time Frequency Transforms for Radar Imaging and Signal Analysis*, Artech House, Boston, 2002.
- [4] Li J., H. Ling and V. C. Chen, "An Algorithm to Detect the Presence of 3D Target Motion from ISAR Data" *Multidimensional Systems and Signal Processing, Special Issue on Radar and Signal Processing and its Applications*, Vol. 14, No. 1, pp. 223-240, January 2003.
- [5] Auger, F., P. Flandrin, P. Gonçalves and O. Lemoine, *Time-Frequency Toolbox – For Use With MatLab*, Centre National de la Recherche Scientifique, France 1996. Available at <http://gdr-isis.org/Applications/tftb/iutsn.univ-nantes.fr/auger/tftb.html> (Date Accessed - 16 October 2003)
- [6] Li, J. and H. Ling, "Use of Genetic Algorithms in ISAR Imaging of Targets with Higher Order Motions," *IEEE Transactions on Aerospace and Electronic Systems* Vol. 39, No. 1, pp. 343 – 351, January 2003.
- [7] Robinson, J. and Y. Rahmat-Samii, "Particle Swarm Optimization in Electromagnetics," *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 2, pp. 397 – 407, February 2004,.
- [8] Thayaparan, T., "MatLab Code for the ISAR Image Focusing using the AJTF Algorithm", Defence Research and Development Canada – Ottawa 2003 (unpublished).
- [9] Goldberg, D.E. *Genetic Algorithms is Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA 1989.
- [10] Chipperfield, A., P. Fleming, H. Pholheim and C. Fonseca, *Genetic Algorithm Tool Box – for Use with MatLab*, Department of Automatic Control and Systems Engineering, University of Sheffield 1999. Available at <http://www.shef.ac.uk/acse/research/ecrg/gat.html> (Date Accessed 6 September 2004).

- [11] Boeringer, D.W. and D. H. Werner, "Particle Swarm Optimization Versus Genetic Algorithms for Phased Array Synthesis," *IEEE Transactions on Antennas and Propagation*, Vol. 52, No 3, pp. 771 – 779, March 2004.
- [12] Thayaparan, T., Scientific Authority, "Time-Frequency Algorithms for Focusing Distorted ISAR Images", Contract Number: W7714-011498/001/SV, Defence Research and Development Canada – Ottawa, 2002. Available at <http://www.drdc-rddc.dnd.ca/> (Date Accessed 5 December 2003)
- [13] The MathWorks, *Image Processing Toolbox – For Use With MatLab*, Version 5.0.2, pp. 8.20-8.31, The Mathworks, Inc, March 2005. Available at <http://www.mathworks.com/access/helpdesk/help/toolbox/images/> (Date Accessed 10 March 2005).
- [14] The MathWorks, *Genetic Algorithm and Direct Search Toolbox – For Use With MatLab*, Version 1.0.3, The Mathworks, Inc, March 2005. Available at <http://www.mathworks.com/access/helpdesk/help/toolbox/gads/> (Date Accessed 10 March 2005).
- [15] Li, J and H. Ling, "Application of Adaptive Chirplet Representation for ISAR Feature Extraction from Targets with Rotating Part," *IEE Proceedings of Radar, Sonar and Navigation*, Vol. 150, No. 4, pp. 284 – 291, August 2003.
- [16] P. Steeghs, L. Kester, and S. Gelsema, "Radon Transforms and Time-Frequency Representations for ISAR Motion Compensation", *Proceedings of SPIE*, Vol. 4738, pp. 252-263, 2002.
- [17] P. Steeghs and S. Gelsema, "Application of Radon Transforms and Time-Frequency Representations to ISAR Imagery", *Proceedings of SPIE*, Vol. 5102, pp. 189-199, 2003.
- [18] Chen, V.C., "B727S.mat Simulated ISAR data with Blurring", U.S. Naval Research Laboratory – Radar Division. Available at <http://airborne.nrl.navy.mil/~vchen/tftsa.html>. (Date Accessed 1 September 2004).

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Ft. Belvoir, Virginia
2. Dudley Knox Library
Naval Postgraduate School
Monterey, California
3. Chairman, Code EC
Electrical and Computer Engineering Department
Monterey, California
4. Michael Morgan, Code/EC/Mw
Electrical and Computer Engineering Department
Monterey, California
5. Thayananthan Thayaparan
Defence Research and Development Canada - Ottawa
Ottawa, Ontario
6. Jeffrey Knorr, Code/EC/Ko
Electrical and Computer Engineering Department
Monterey, California
7. Wade Brinkman
Canadian Forces
Ottawa, Ontario